Universidad de las Ciencias Informáticas

"Facultad 6"

Sistema de Información Geográfica para el Ministerio de Educación Superior

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

Autor: Luis José Tristá Rodríguez

Tutor: Ing. Pedro Enrique Palau Isaac

La Habana, Junio del 2012

"Año 54 de la Revolución"

"Toda la gloria del mundo cabe en un grano de maíz…"

José Martí.

Declaración de autoría

Declaro ser autor de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

_____ _____

Luis José Tristá Rodríguez Ing. Pedro Enrique Palau Isaac

Datos de Contacto

Tutor: Ing. Pedro Enrique Palau Isaac

Formación Académica: Ingeniero en Ciencias Informáticas.

Centro Laboral: Universidad de las Ciencias Informáticas (UCI).

Correo electrónico: pepalau@uci.cu

Dedicatoria

A la persona más importante en mi vida, mi madre, por apoyarme siempre aunque todo esté en mi contra. Por su confianza, amor y cariño.

A mi padre por ser ejemplo de hombre y amigo.

A Mirtha y Guirola, mis segundos padres.

A Yordi mi hermano y guía incondicional, esto es para ti…

A mis abuelos, por ser una fuente inagotable de conocimientos y experiencia. A Tristá por las largas horas de conversaciones y consejos. A Tato por su paciencia.

A mis tíos por su preocupación.

A toda mi familia por estar siempre pendientes de mí, a los cuales amo y aprecio más que a nada en el mundo.

A mis amigos por enseñarme lo hermoso que es la amistad.

A todas las personas que luchan por lograr un objetivo.

A ustedes, lectores de estas páginas…

Agradecimientos

Agradezco a toda mi familia por apoyarme durante toda mi vida, especialmente a mis padres y abuelos.

Agradezco además a mis grandes amigos de Santa Clara y la UCI que son como mi familia.

Gracias a Anita por su compañía y ayuda incondicional en los momentos difíciles.

Gracias a todos aquellos que me brindaron su mano durante el transcurso de este trabajo, Adrián, Yenier, Alain, Alejandro, Membrides y los muchachos de Aplicativos SIG.

Gracias a Pedro, mi tutor, por su preocupación y guía certera.

Gracias a todos los que de una forma u otra forman parte de este trabajo.

Resumen

Los Sistemas de Información Geográfica (SIG) y el análisis estadístico son dos campos de trabajo que comparten muchos intereses. Los SIG aplicados a procesos de toma de decisiones a partir de datos de diferentes tipos han adquirido un gran auge a nivel mundial. Este tipo de sistemas aportan un conjunto de instrumentos que pueden ser aplicados en diferentes ramas de la ciencia y, que a su vez facilitan la evaluación de indicadores de importancia. En el Ministerio de Educación Superior (MES) de la República de Cuba existen grandes dificultades en el proceso de análisis y evaluación de la información de los centros universitarios, debido a que la institución no cuenta con un sistema informático que le brinde las herramientas para la evaluación de la información asociada a dichos centros. En investigaciones anteriores se ha analizado y diseñado una solución informática para resolver dicho problema. El presente trabajo describe el proceso de implementación de un SIG para la representación geográfica de los centros universitarios y la gestión de la información asociada a ellos. Además, se identifica y analiza la situación problemática, y se materializan los resultados de investigaciones anteriores acoplados a la presente investigación. Se realiza un estudio de las diferentes herramientas y tecnologías seleccionadas, para el modelado y desarrollo de la herramienta y finalmente se procede con la implementación del sistema propuesto.

Palabras claves: Análisis, implementación, indicadores, MES, SIG.

Índice de Tablas

Tabla 1 Sección a probar en Insertar centro universitario 36
Tabla 2 Descripción de las variables en Insertar centro universitario 37
Tabla 3 Matriz de datos en Insertar centro universitario 39
Tabla 4 Sección a probar en Crear gráfico por centro universitario 40
Tabla 5 Descripción de las variables en Crear gráfico por centro universitario 41
Tabla 6 Matriz de datos Crear gráfico por centro universitario 41
Tabla 7 Sección a probar en Crear gráfico por provincia 42
Tabla 8 Descripción de las variables en Crear gráfico por provincia 43
Tabla 9 Matriz de datos Crear gráfico por provincia 43

Índice de Figuras

Figura 1 Tecnologías que conforman AJAX 15
Figura 2 Flujos de trabajo y fases de RUP 21
Figura 3 Diagrama de Casos de Uso del Sistema 24
Figura 4 Diagrama de Componentes caso de uso “Crear Mapa Temático” 27
Figura 5 Diagrama de Despliegue: SIG-MES 27

Índice

Introducción 1

Capítulo 1 5

1.1 Introducción 5
1.2 Análisis de la situación problemática 5
1.3 Conceptos asociados a la investigación 6
1.4 Tendencias actuales de los Sistemas de Información Geográfica 7
1.5 Paradigmas de programación 8
1.6 Tecnologías 10
1.6.1 Arquitectura Cliente/Servidor 10
1.6.2 Lenguajes de Programación 10
1.6.2.1 Lenguajes de programación del lado del cliente 11
1.6.2.1.1 HTML 11
1.6.2.1.2 CSS 12
1.6.2.1.3 JavaScript 12
1.6.2.2 Lenguajes de programación del lado del servidor 13
1.6.2.2.1 PHP 13
1.6.3 Técnicas y Formatos de intercambio de datos 13
1.6.3.1 JSON 13
1.6.3.2 AJAX 14
1.6.4 Plataformas, herramientas y frameworks de desarrollo 15
1.6.4.1 ExtJS 15
1.6.4.2 CartoWeb 15
1.6.4.3 GeneSIG 16
1.6.4.4 Servidor Web Apache 17
1.6.4.5 Servidor de Mapas MapServer 17
1.6.4.6 NetBeans como IDE para PHP 17
1.6.4.7 Visual Paradigm como Herramienta Case 18
1.6.5 Sistemas Gestores de Bases de Datos 18
1.6.5.1 PostgreSQL 18
1.6.5.2 PostGIS 19
1.7 Metodologías de Desarrollo 19
1.7.1 RUP 20
1.7.1.1 Lenguaje Unificado de Modelado (UML) 22

1.8 Conclusiones parciales 22
Capítulo 2 23
2.1 Introducción 23
2.2 Valoración crítica sobre el análisis propuesto. 23
2.3 Requisitos funcionales. 24
2.4 Generalidades de la implementación. 26
2.4.1 Modelo de Implementación. 26
2.4.1.1 Diagrama de componentes. 26
2.4.1.2 Modelo de Despliegue. 27
2.5 Estilos de programación. 28
2.6 Estándares de codificación. 30
2.7 Conclusiones parciales 31
Capítulo 3 32
3.1 Introducción 32
3.2 Descripción general de las pruebas. 32
3.3 Pruebas de Caja Negra 33
3.4 Diseño de las pruebas para validar la solución propuesta 34
3.5 Conclusiones parciales 43
Conclusiones 45
Recomendaciones 46
Referencias Bibliográficas 47
Bibliografías Consultadas 49
Anexos 50
Anexo 1: Diagramas de componentes. 50
Glosario de Términos 56

Introducción

Numerosos cambios tecnológicos y sociales han caracterizado estos últimos años. Las nuevas Tecnologías de la Información y las Comunicaciones se han apoderado de todas las ramas de las ciencias, jugando un papel importante en la sociedad y formando parte de todas las actividades realizadas por los seres humanos. El desarrollo es innegable en los tiempos actuales, han aparecido nuevas ideas, conceptos y teorías desconocidas e irrealizables. Los nuevos métodos de comunicación han cambiado la forma de pensar y de manifestarnos, los obsoletos sistemas de cómputo han quedado reemplazados por nuevos sistemas informáticos, y han provocado nuevas tendencias sociales, económicas y culturales.

En Cuba, la informatización de la sociedad se define como el proceso de utilización ordenada y masiva de las TIC[1] para satisfacer las necesidades de información y conocimiento de todas las personas y esferas de la sociedad. Este proceso busca lograr más eficacia y eficiencia, que permitan una mayor generación de riquezas y hagan sustentable el aumento sistemático de la calidad de vida de los cubanos. (MINREX, 2011)

La demanda de información ha crecido exponencialmente, así como la necesidad de innovación para resolver innumerables problemáticas, esto ha incidido de forma directa en el desarrollo de nuevas tecnologías, incluyendo a los Sistemas de Información Geográfica (SIG), permitiendo manipular, analizar y gestionar datos importantes, aplicables a varias disciplinas de la investigación científica.

Un SIG es un sistema de hardware, software y procedimientos, diseñados para soportar la captura, el manejo, la manipulación, el análisis, el modelado y el despliegue de datos espacialmente referenciados (geo-referenciados), para la solución de problemas complejos del manejo y planeamiento territorial (Silva, 2005).

En el siglo XXI el desarrollo de los SIG ha aumentado considerablemente, pero su correcto desempeño se reducía a un pequeño número de plataformas. Los usuarios comenzaron a exportar el concepto de visualización de datos SIG a Internet. Surge la necesidad de utilizar Sistemas de Información Geográfica de código abierto, y así flexibilizar futuros cambios y mejoras. Estos sistemas, a diferencia del software privativo, suelen abarcar una gama más amplia de plataformas, permitiendo su modificación y especialización en tareas específicas para la resolución de un problema dado. Su evolución ha tenido un desarrollo considerable y son prácticamente indispensables ya que ayudan al

[1] TIC (Tecnologías de la Información y las Comunicaciones)

proceso de toma de decisiones en diferentes circunstancias. Permitiendo relacionar información de cualquier tipo con una posición geográfica.

El Ministerio de Educación Superior es el encargado de gestionar los datos referentes a los centros universitarios del país, los cuales se consultan y analizan manualmente usando métodos convencionales de comunicación, por lo que se les dificulta el trabajo a las personas encargadas de manipular estos datos. De esta forma se malgasta tiempo y recursos. En la universidad de las Ciencias Informáticas (UCI) se llevó a cabo una investigación para solucionar la problemática identificada, esta propuso el desarrollo de un Sistema de Información Geográfica (SIG) como la herramienta necesaria para acelerar el análisis y la gestión de la información, permitiendo la representación geoespacial de dicha información. Hasta la actualidad no se han completado todas las fases de desarrollo de la solución informática, solamente las fases de análisis y diseño, sin lograr la implementación. Durante esta fase el programador juega un papel fundamental, ya que es el encargado de definir y mantener el código fuente, materializando el diseño propuesto.

Debido a lo anteriormente mencionado, se deriva entonces el siguiente **problema a resolver:** *¿Cómo lograr la funcionalidad para diseño propuesto del Sistema de Información Geográfica del Ministerio de la Educación Superior?* Para dar solución al mismo, el **objeto de estudio** se define como el *proceso de implementación de los Sistemas de Información Geográfica*, siendo el **campo de acción** la *implementación del Sistema de Información Geográfica del Ministerio de la Educación Superior.* Como **objetivo general** se pretende *elaborar la documentación técnica de la implementación del Sistema de Información Geográfica del Ministerio de la Educación Superior*. Por lo que se expone la siguiente **idea a defender:** *Con la elaboración de la documentación técnica correspondiente a la implementación del diseño propuesto, se obtendrá un Sistema de Información Geográfica para el Ministerio de la Educación Superior*.

Para lograr el objetivo expuesto anteriormente se desarrollarán las siguientes tareas:

1. Realización del marco teórico de la investigación.
2. Fundamentar tendencias actuales, tecnologías y conceptos más importantes relacionados con el desarrollo de Sistemas de Información Geográfica.
3. Estudiar, analizar y modificar el diseño propuesto del sistema para su implementación.
4. Implementación del Sistema de Información Geográfica.

5. Validación del Sistema Informático.

Una vez realizadas las tareas de investigación se espera el siguiente resultado:

- La documentación técnica producto del desarrollo del Rol de Implementador del Sistema de Información Geográfica para el Ministerio de la Educación Superior.

Toda investigación científica utiliza métodos científicos para la investigación debido a que son: la forma de abordar la realidad, de estudiar la naturaleza, la sociedad y el pensamiento, con el propósito de descubrir su esencia y sus relaciones (Rolando Alfredo Hernández León, 2002). Dichos métodos científicos se clasifican fundamentalmente en:

1. **Métodos Empíricos:** Permiten la obtención y elaboración de los datos empíricos y el conocimiento de los hechos fundamentales que caracterizan los fenómenos (Rolando Alfredo Hernández León, 2002).
   - **Observación:** Se realizarán varias visitas al Ministerio de Educación Superior (MES), con el objetivo de observar los procesos y mecanismos que allí se desarrollan para la manipulación de los datos en los centros universitarios.
2. **Métodos Teóricos:** Se utilizan en la construcción y desarrollo de la teoría científica y en el enfoque general para abordar los problemas de las ciencias (Rolando Alfredo Hernández León, 2002).
   - **Analítico-Sintético:** Para identificar y analizar los elementos y términos más importantes dentro del campo de los Sistemas de Información Geográfica.
   - **Histórico-Lógico:** Para realizar el análisis de los conceptos y evolución de soluciones existentes relacionadas con los Sistemas de Información Geográfica.
   - **Modelación:** Este método se utilizó para la modelación de diagramas teniendo en cuenta la metodología de desarrollo de software seleccionada.

La presente investigación está dividida en 3 capítulos:

Capítulo 1: Fundamentación Teórica. Se abordará la situación problemática y las tecnologías existentes que se emplearán para llevar a cabo la solución propuesta, serán descritas las principales características de las plataformas y frameworks que se utilizarán, así como las herramientas de desarrollo.

Capítulo 2: Análisis y construcción de la solución propuesta. En dicho capítulo se hará una valoración de la solución propuesta, para mejor entendimiento y desempeño de los desarrolladores durante la implementación del sistema.

Capítulo 3: Validación de la solución propuesta. En el mismo se analiza la solución dada, se comprobará que cumpla los requerimientos propuestos.

Capítulo 1

Fundamentación Teórica

1.1 Introducción

En el presente capítulo se analiza en profundidad la situación problemática. Se explican las herramientas, tecnologías y técnicas, así como los conceptos relacionados con estas, para llevar a cabo la construcción de la solución planteada, además, de paradigmas y lenguajes de programación utilizados en la investigación.

1.2 Análisis de la situación problemática

El Ministerio de Educación Superior (MES), posee una red de 68 centros universitarios, de los cuales maneja toda su información. Para gestionar los datos de estos centros, es necesario hacerlo por vías tradicionales como teléfono, o dirigirse a cada uno de ellos, solicitarla por los antiguos métodos de comunicación, correo postal, o haciendo uso de correo electrónico o fax, de igual forma es un trabajo muy complejo debido al volumen de información que se manipula. Al personal encargado de interactuar con los datos de interés para el MES se le dificulta el trabajo con la información asociada a estos centros, ya que es un proceso complejo y muy lento manualmente, retrasando la obtención inmediata de resultados, poniendo en riesgo la exactitud de las valoraciones y siendo un obstáculo para la toma de decisiones, malgastándose tiempo necesario y recursos materiales.

Debido a esta situación, aún vigente en el MES, en la Universidad de las Ciencias Informáticas (UCI), en cumplimiento a uno de sus principales objetivos (colaborar con la informatización de la sociedad cubana y de sus principales instituciones) se desarrolló una investigación que buscaba como principal resultado solucionar la problemática identificada. Dicha investigación propuso el desarrollo de un Sistema de Información Geográfica (SIG) como la herramienta capaz, tanto de acelerar el análisis y la gestión de la información de los centros universitarios, así como aumentar las expectativas de trabajo relacionado con estos procesos. A diferencia de otro tipo de sistema informático, un SIG permite ampliar considerablemente los datos hasta su representación geográfica o geoespacial, lo que le facilitará al MES comprender mejor los estudios complejos que desee realizar sobre dicha información, estructurar y organizar los resultados de los análisis por niveles organizativos y políticos más

profundos (región, municipio, provincia y país). Se podrá evaluar el comportamiento de variables e indicadores por rangos de selección en esos niveles, dados sus valores, además de examinar significativamente la representación o visualización de los datos así como su socialización con el resto de los organismos nacionales y el estado.

Todos estos aspectos hacen de un SIG la mejor propuesta de solución, pero como sistema informático requiere de una metodología que guíe todo el proceso de desarrollo y especifique los roles que deben dar cumplimento a los pasos (fases de desarrollo) que plantee dicha metodología. La investigación mencionada anteriormente no llevó a cabo la realización de todas esas fases para obtener el producto final. Completó solamente las fases de Análisis y Diseño del software propuesto, por lo que aún no cumple con su objetivo principal. Durante el tiempo definido se estudiaron y analizaron todos los aspectos y particularidades del negocio, se determinaron las características funcionales o de funcionamiento que debía tener el sistema (requerimientos funcionales y no funcionales) y se modelaron los principales diagramas para lograr su funcionalidad (Diagrama de Casos de Uso del Sistema, Diagrama de Clases del Diseño, etc.), pero no se logró iniciar la fase de Construcción. De esta fase, los subprocesos o hitos de desarrollo deben ser llevados a cabo por el rol de programador o implementador, quien juega un papel importante en el desarrollo de software ya que es el encargado de definir y mantener el código fuente, materializando así el diseño propuesto por el diseñador de un sistema informático. El implementador es responsable además de los componentes de desarrollo y de prueba, de acuerdo con los estándares aprobados por el proyecto para la integración en subsistemas más grandes.

1.3 Conceptos asociados a la investigación

Ministerio de Educación Superior: Es la institución encargada de dirigir la Educación Superior en Cuba, creada con el objetivo de aplicar la política educacional en el nivel de la enseñanza superior y dirigirla metodológicamente. Este Ministerio tiene 68 instituciones de nivel superior que incluye 3150 sedes universitarias municipales.

Sistema de Información Geográfica: Un Sistema de Información Geográfica es una integración organizada de hardware, software y datos geográficos diseñada para capturar, almacenar, manipular, analizar y desplegar en todas sus formas la información geográficamente referenciada con el fin de resolver problemas complejos de planificación y gestión (Thompson, 2008).

Datos espaciales: Colección de datos orientados a la geografía referenciados en el espacio.

Mapa: Es una representación gráfica y métrica de una porción de territorio generalmente sobre una superficie bidimensional pero que puede ser también esférica. Un mapa que tenga propiedades métricas significa que ha de ser posible tomar medidas de distancias, ángulos o superficies sobre él y obtener un resultado aproximadamente exacto. Los mapas utilizan dos tipos de formatos, el ráster y el vectorial.

Información: Es un conjunto de datos acerca de algún suceso, hecho o fenómeno, que organizados en un contexto determinado tienen su significado, cuyo propósito puede ser el de reducir la incertidumbre o incrementar el conocimiento acerca de algo (Thompson, 2008).

Información geográfica: Es un conjunto de datos espaciales, los cuales brindan una información de algún hecho o fenómeno (Thompson, 2008).

Cartografía digital: La cartografía digital es el procedimiento que transforma la información geográfica de los mapas de papel a coordenadas digitales. Es la ciencia que se encarga del estudio y elaboración de los mapas geográficos y territoriales de diferentes dimensiones lineales. De acuerdo a los mapas básicos, el campo de la cartografía se puede dividir o separar en dos categorías generales: la Cartografía general y la Cartografía temática (Rico, 2000).

La cartografía general: Implica los mapas que se construyan para una audiencia general y contengan así una variedad de características. Se utiliza para representar áreas del terreno que muestran los elementos naturales (curvas de nivel, aguas, red hídrica), elementos artificiales, humanos o culturales, como son las redes de transporte y los centros poblados. También muestran fronteras políticas, como pueden ser los límites de las ciudades, de los municipios o de los departamentos (Rico, 2000).

La cartografía temática: Es la rama de la cartografía que es utilizada por otras ciencias para representar gráficamente sobre un plano los objetos y fenómenos del universo. Implican los mapas de temas geográficos específicos, orientados hacia audiencias específicas. La finalidad de la cartografía temática es representar a través de actividades técnicas, científicas, tecnológicas y artísticas el mundo real, los fenómenos y los objetos del universo sobre un plano (Rico, 2000).

1.4 Tendencias actuales de los Sistemas de Información Geográfica

En la actualidad, la complejidad, cantidad y tratamiento de la información geográfica ha alcanzado un auge palpable a escala mundial. Debido a ello, se han desarrollado herramientas avanzadas para la gestión de dicha información como soluciones a los diferentes problemas que se identifican en

disímiles ramas de la sociedad, destacándose en este aspecto los Sistemas de Información Geográfica (SIG). Según estudios internacionales, se demostró que más del 80% de toda la información involucrada en la toma de decisiones de los gobiernos e instituciones empresariales, posee un componente geoespacial. Basado en esta estimación, es que se ha extendido el uso de los SIG para acelerar y colaborar con este proceso. Muestra de ello, es el uso que se les da a estos sistemas en el control de los activos territorialmente distribuidos de las entidades, y en la asistencia que brindan a ciudadanos en sus trámites o necesidades informativas cotidianas.

Cuba, y específicamente en la Universidad de las Ciencias Informáticas, no se descartan las ventajas de estas aplicaciones informáticas. Es por ello que todas las expectativas creadas sobre los SIG están presentes a pesar de sus correspondientes limitaciones. De la gran cantidad de estos sistemas que existen en estos momentos a nivel mundial, solamente unos pocos son conocidos y utilizados en el país. Existen SIG creados y desarrollados por empresas españolas, holandesas, brasileñas, y de muchos otros países que pueden adquirirse con sus licencias. Pero la gran parte de los especialistas cubanos prefieren los productos de las compañías de EE.UU, que se dedican al desarrollo de software para SIG, por las funcionalidades y la calidad presente en estos. Los sistemas diseñados y creados en Estados Unidos, que son adquiridos en Cuba por distintas vías, no tienen licencia de utilización de usuario debido a que las compañías estadounidenses no están autorizadas a vender este tipo de software a los cubanos (Silva, 2005). La Universidad de las Ciencias Informáticas quiere convertirse en modelo de ciudad universitaria digital y de sociedad de información en Cuba (Fung, 2010). En dicha universidad se han desarrollado algunos Sistemas de Información Geográfica, con diferentes fines y para variadas ramas de la sociedad, brindando a los clientes información importante dependiendo del sistema consultado.

1.5 Paradigmas de programación

Los paradigmas de programación representan un enfoque para el desarrollo de un software. Existen criterios diferentes y comparaciones, pero según las características de cada paradigma los hacen ajustarse a ciertos problemas, tienen ventajas y desventajas, las cuáles los diferencian. Los paradigmas de programación más comunes son los siguientes:

Paradigmas procedimentales u operacionales, es tal vez el más conocido y utilizado en el proceso de programación, donde los programas se desarrollan a través de procedimientos. Pascal C y BASIC son tres de los lenguajes imperativos más importantes. La palabra latina imperare significa "dar instrucciones". El paradigma se inició al principio del año 1950 cuando los diseñadores reconocieron

que las variables y los comandos o instrucciones de asignación constituían una simple pero útil abstracción del acceso a memoria y actualización del conjunto de instrucciones máquina. Debido a la estrecha relación con la arquitectura de la máquina, los lenguajes de programación imperativa pueden ser implementados muy eficientemente, al menos en principio. (San Marcos, 2008)

El paradigma declarativo o paradigma de programación lógica se basa en el hecho que un programa implementa una relación antes que una correspondencia. Debido a que las relaciones son más generales que las correspondencias (identificador - dirección de memoria), la programación lógica es potencialmente de más alto nivel que la programación funcional o la imperativa. El lenguaje más popular enmarcado dentro de este paradigma es el lenguaje PROLOG. El auge del paradigma declarativo se debe a que el área de la lógica formal de las matemáticas ofrece un sencillo algoritmo de resolución de problemas adecuado para, usarse en un sistema de programación declarativo de propósito general. (San Marcos, 2008)

La programación funcional se caracteriza por el uso de expresiones y funciones. Un programa dentro del paradigma funcional, es una función o un grupo de funciones compuestas por funciones más simples estableciéndose que una función puede llamar a otra, o el resultado de una función puede ser usado como argumento de otra función. El lenguaje por excelencia ubicado dentro de este paradigma es el LISP[2]. (San Marcos, 2008)

El paradigma orientado a objetos, se basa en los conceptos de objetos y clases de objetos. Un objeto es una variable equipada con un conjunto de operaciones que le pertenecen o están definidas para ellos. El paradigma orientado a objetos actualmente es el paradigma más popular. Una de las bondades importantes de los lenguajes orientados a objetos es que las definiciones de los objetos pueden usarse una y otra vez para construir múltiples objetos con las mismas propiedades o modificarse para construir nuevos objetos con propiedades similares pero no exactamente iguales. (San Marcos, 2008)

La Programación Orientada a Objetos (POO) no es un lenguaje de programación, puede aplicarse a cualquier lenguaje, y de hecho hoy en día está disponible en mayor o menor medida en todos los lenguajes tradicionales (C se ha convertido en C++, Pascal en Delphi, Visual Basic incorpora parte de la POO) y no aparece un lenguaje nuevo sin que incluya POO (como es el caso de Java). (Morero, 1999-2000)

[2] **LIS**t **P**rocessing (Proceso de listas). Familia de lenguajes de programación de tipo multiparadigma.

1.6 Tecnologías

El SIG, desarrollado sobre un ambiente Web, emplea un conjunto de tecnologías las cuales son de vital importancia para su correcto funcionamiento. Desde el punto de vista del implementador es necesario su estudio y conocimiento en profundidad, para obtener el máximo rendimiento además de explotar las ventajas que ofrecen.

1.6.1 Arquitectura Cliente/Servidor

Las arquitecturas cliente/servidor dominan el horizonte de los sistemas basados en computadora. Todo existe: desde redes de cajeros automáticos hasta Internet, y esto es debido a que el software que reside en una computadora (el cliente) solicita servicios y/o datos de otra computadora (el servidor) (Pressman, 2005).

En esta arquitectura la capacidad de proceso está repartida entre los clientes y los servidores, aunque resaltan las características de tipo organizativas ya que se centraliza la gestión de la información y se separan las responsabilidades en el sistema, por ejemplo, puede existir un servidor Web en un nodo de la red y un servidor de bases de datos en otro. Esta separación es de tipo lógico, donde el servidor no reside solamente en una máquina ni es necesariamente un solo programa, la arquitectura básica seguirá siendo la misma. Los clientes están conectados a un servidor, en el que se centralizan recursos y aplicaciones, que se pone a disposición de los clientes cada vez que estos lo solicitan. Esto significa que todas las gestiones que se realizan se concentran en el servidor, de manera que en él, se disponen los requerimientos provenientes de los clientes, los archivos que son de uso público y los que son de uso restringido. Comúnmente en los sistemas multicapa el servidor se descompone en diferentes programas que pueden estar alojados en diferentes ordenadores, aumentando la eficiencia y el grado de distribución del sistema. Esta arquitectura es un pilar fundamental del sistema que se propone, este como aplicación Web, se desarrolla bajo estos conceptos.

1.6.2 Lenguajes de Programación

Las máquinas de cómputo, necesitan un lenguaje el cuál interpretan para llevar a cabo instrucciones, mediante las cuales se controla su comportamiento y se realizan diferentes operaciones. Los lenguajes de programación son los encargados de relacionar a los seres humanos con los ordenadores, y así manipularlos para automatizar tareas, algunos de ellos son Pascal, Basic, Perl, Python, C++, Java y

PHP[3]. El lenguaje de programación está conformado por una serie de reglas sintácticas y semánticas que son utilizadas por el programador y a través de las cuales creará un programa o subprograma. Todas estas instrucciones escritas por el desarrollador y que forman dicho programa es a lo que se denomina código fuente (Lanzillotta, 2005).

Los lenguajes de programación pueden ser interpretados o compilados. Los interpretados, usan otro programa (Intérprete) para traducir las instrucciones a código máquina. Por lo general son más lentos que los compilados ya que necesitan un intermediario entre él y la máquina, pero ganan en portabilidad ya que haciendo uso de su intérprete pueden ejecutarse en múltiples plataformas. Los lenguajes compilados solo necesitan un compilador que traduzca el código a código máquina, generando instrucciones compiladas que se ejecutan directamente en el ordenador, suelen ser mucho más rápidos en tiempo de ejecución.

1.6.2.1 Lenguajes de programación del lado del cliente

1.6.2.1.1 HTML

HTML (Hypertext Markup Language), Lenguaje de Marcado de Hipertexto, usado para escribir y publicar documentos en la World Wide Web (WWW[4]). Es una aplicación de la ISO[5] Standard 8879:1986 (SGML, Standard Generalized Markup Language) (HTML W3C, 2000). Es el lenguaje con el que se escriben todas las páginas Web en Internet. Este lenguaje fue originalmente concebido para el intercambio de documentos de corte científicos y técnicos. HTML utiliza etiquetas, que consisten en breves instrucciones de comienzo y final, mediante las cuales se determina la forma en la que debe mostrar el contenido en un navegador, imágenes, textos entre otras aplicaciones. Tiene como todo lenguaje sus propias reglas, las etiquetas se identifican porque se encierran entre signos “menor que” y “mayor que”, además de sus atributos. El sistema implementado hace uso de este lenguaje, como todas las aplicaciones Web necesita que los datos enviados al navegador del cliente posean este formato para poder ser mostrados correctamente. HTML se ha convertido en un estándar de la World Wide Web Consortium (W3C), por lo que puede ser visualizado en múltiples navegadores y diferentes sistemas operativos.

[3] Hypertext Preprocessor (Preprocesador de Hipertexto).
[4] World Wide Web (Red Informática Mundial).
[5] International Organization for Standardization (Organización Internacional de Normalización).

1.6.2.1.2 CSS

CSS (Cascading Style Sheets), Hojas de Estilo en Cascada, se usa para definir la presentación de un documento escrito en HTML o XML[6], por extensión XHTML[7] (CSS W3C, 2000). W3C es el encargado de formular las reglas CSS, que servirán de estándar para la correcta visualización en los navegadores de los clientes. Principalmente es usado para separar la estructura de los documentos HTML de su presentación, de esta forma lograr mejor organización en la estructura de código y de directorios de los sitios Web. Cuando es utilizado CSS, la estructura de código HTML no debe proporcionar información de cómo va a ser mostrada visualmente la etiqueta, esta información debe formar parte del fichero CSS que se debe encontrar separado del HTML. Se pueden definir varias especificaciones entre las que se encuentran color de texto, color de fondo y borde. La información de estilo puede ser adjuntada como un fichero separado o en el mismo documento HTML. En este último caso podrían definirse estilos en la cabecera del documento o en cada etiqueta particular mediante el atributo "style". Su uso es indispensable en sistemas de entorno Web como el que se expone en la presente investigación, ya que a través de CSS la presentación es centralizada, de forma que se puede agilizar la actualización de un sitio Web completo en cuanto a estilo e interfaz visual. Es necesario señalar la importancia de la separación de la presentación del código HTML, resultando más fácil de entender y reduciéndose considerablemente el tamaño de los ficheros y por consiguiente aumentado la velocidad de carga de las páginas Web.

1.6.2.1.3 JavaScript

JavaScript es un lenguaje de programación interpretado. Basado en prototipos, imperativo, débilmente tipado y dinámico. Se utiliza principalmente en su forma del lado del cliente (client-side), implementado como parte de un navegador Web permitiendo mejoras en la interfaz de usuario y páginas Web dinámicas, aunque existe una forma de JavaScript del lado del servidor (Server-side JavaScript o SSJS). También es usado en aplicaciones externas a la Web, por ejemplo en documentos PDF[8] y aplicaciones de escritorio. JavaScript se diseñó con una sintaxis similar al C, aunque adopta nombres y convenciones del lenguaje de programación Java.(Ecured, 2011).

Todos los navegadores modernos interpretan el código JavaScript integrado en las páginas Web, aunque no de la misma forma. Para interactuar con una página Web se provee al lenguaje JavaScript

[6] Extensible Markup Language (Lenguaje de marcas extensible).
[7] Extensible HyperText Markup Language (HTML expresado como XML).
[8] Portable Document Format (Formato de Documento Portátil).

de una implementación del Document Object Model (DOM).Tradicionalmente se venía utilizando en páginas HTML para realizar operaciones y únicamente en el marco de la aplicación cliente, sin acceso a funciones del servidor. JavaScript se interpreta en el agente de usuario, al mismo tiempo que las sentencias van descargándose junto con el código HTML. GeneSIG, la plataforma seleccionada para la construcción del Sistema de Información Geográfica, hace uso intensivo de este lenguaje del lado del cliente, permitiendo enriquecer la presentación de los datos, la experiencia del usuario y la optimización de la comunicación con el servidor. Estas tareas serán desarrolladas con el uso de la librería ExtJS, la cual será explicada en epígrafes posteriores.

1.6.2.2 Lenguajes de programación del lado del servidor

1.6.2.2.1 PHP

PHP (Hypertext Preprocessor), aunque surge con el nombre Personal Home Page, es un lenguaje de programación interpretado que va embebido (incrustado) en páginas HTML. Su sintaxis es similar a la utilizada en otros lenguajes de programación de alto nivel como C, Java y Perl, sólo con algunas diferencias. El principal objetivo del lenguaje es permitir a los desarrolladores Web codificar rápidamente páginas que se generan dinámicamente. Es un lenguaje del lado del servidor, que responde a peticiones de un cliente, el servidor Web, con soporte para PHP, interpreta la petición y envía una respuesta al usuario. PHP es de código abierto, apoyado por una amplia comunidad que se dedica al continuo desarrollo y fortalecimiento del lenguaje (PHP, 2011). Del lado del servidor, GeneSIG está programado en PHP 5 el cual está enfocado al desarrollo de aplicaciones Web en el mismo lenguaje de programación. La versión mínima de PHP requerida para ejecutar GeneSIG es PHP 5.2. Se considera indispensable para la implementación del SIG-MES, por ser el lenguaje base que utiliza GeneSIG, es multiplataforma, permite aplicar técnicas de programación orientada a objetos, soporta conexión con la mayoría de los motores de base de datos que se utilizan en la actualidad además de ser libre, lo cual apoya las políticas de migración del país.

1.6.3 Técnicas y Formatos de intercambio de datos

1.6.3.1 JSON

JSON (JavaScript Object Notation), Notación de Objetos de JavaScript, es un formato ligero de intercambio de datos. Para los seres humanos es fácil leerlo e interpretarlo, al igual que para las máquinas generarlos. Es un formato de texto que es completamente independiente del lenguaje pero utiliza convenciones que son ampliamente conocidas por los programadores de la familia de lenguajes

C, incluyendo C, C++, C#, Java, JavaScript, Perl, Python, y muchos otros. Estas propiedades hacen que JSON sea un lenguaje ideal para el intercambio de datos (Json, 2011). La comunicación en el sistema entre el cliente y el servidor se realiza utilizando el formato JSON, y de esta forma se unifican los lenguajes de programación JavaScript y PHP para dar solución a diferentes tareas, cada uno en el contexto que se desarrolla. La claridad y simplicidad para manipular los datos transferidos son características fundamentales a tener en cuenta en el desarrollo de aplicaciones Web, por lo cual es seleccionado.

1.6.3.2 AJAX

AJAX (Asynchronous JavaScript and XML), JavaScript Asíncrono y XML (eXtensible Markup Language). Estas aplicaciones se ejecutan en el cliente, es decir, en el navegador de los usuarios mientras se mantiene la comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre las páginas sin necesidad de recargarlas, lo que significa aumentar la interactividad, velocidad y usabilidad en las aplicaciones (Ecured, 2011). Es una técnica muy usada en el desarrollo de webs interactivas, mediante la combinación de tecnologías ya existentes. No interfiere en la visualización ni el comportamiento de la página, pero solo es soportado por navegadores modernos. La Web 2.0[9] hace un gran uso de esta técnica, aumentando grandemente el rendimiento de las páginas y mejorando la experiencia de los usuarios. La utilización de AJAX provee a GeneSIG de mecanismos para la comunicación asíncrona, brindando a sus componentes un recurso valioso mediante funcionalidades ya implementadas para realizar peticiones al servidor, de forma transparente y fácil de utilizar. Sin su uso la plataforma estaría atada a los antiguos métodos de comunicación de la Web y no se lograría el nivel de interactividad y usabilidad que presenta hoy.

A continuación se listan las tecnologías que conforman AJAX.

1. HTML (o XHTML) y CSS para presentar la información.

2. DOM para interactuar dinámicamente con la presentación.

3. XML, XSLT [10] y JSON para intercambiar y manipular datos de manera asíncrona con el servidor Web.

[9] Aplicaciones web que facilitan compartir información, interoperabilidad, diseño centrado en el usuario y colaboración en la World Wide Web.

[10] Extensible Stylesheet Language Transformations (Transformaciones de lenguajes extensibles de hojas de estilo)

4. XMLHttpRequest para el intercambio asíncrono de información.

5. JavaScript para unir todas las tecnologías anteriores.

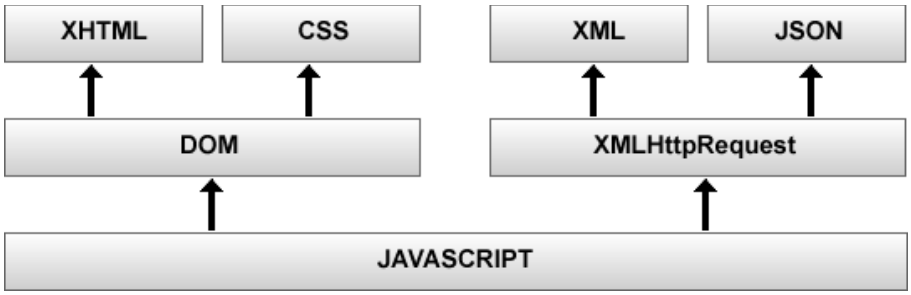

Figura 1 Tecnologías que conforman AJAX

1.6.4 Plataformas, herramientas y frameworks de desarrollo

En el desarrollo de software, un framework simplifica el desarrollo de una aplicación mediante la automatización de algunos de los patrones utilizados para resolver las tareas comunes. Además, un framework proporciona estructura al código fuente, forzando al desarrollador a crear código más legible y más fácil de mantener. Por último, un framework facilita la programación de aplicaciones, ya que encapsula operaciones complejas en instrucciones sencillas (Potencier, 2008).

1.6.4.1 ExtJS

ExtJS es una librería JavaScript que permite construir aplicaciones complejas en internet además de flexibilizar el manejo de componentes de la página como el DOM, peticiones AJAX y DHTML. Tiene la gran funcionalidad de crear interfaces de usuario bastante funcionales (Ecured, 2011). Es compatible con los navegadores más usados a nivel mundial y fundamental dentro de la plataforma GeneSIG, usada en la construcción de componentes para interfaces de usuario de gran rendimiento, altamente personalizables y responsable de casi toda la programación del lado del cliente. Garantiza un balance entre cliente y el servidor. La carga de procesamiento se distribuye, permitiendo que el servidor, al tener menor carga, pueda manejar más clientes al mismo tiempo. Gestiona la comunicación asíncrona y su uso es relativamente fácil.

1.6.4.2 CartoWeb

CartoWeb es un framework para el desarrollo de Sistemas de Información Geográfica de entorno Web, adecuado para la creación de aplicaciones avanzadas y altamente personalizable. Desarrollado por

Camptocamp S.A[11], distribuido bajo licencia GPL[12] y escrito en PHP5. Posee una estructura modular y adaptable a su arquitectura orientada a objetos. Se ejecuta de manera uniforme en Windows o plataformas Unix y muestra su verdadero poder cuando está asociado a PostgreSQL / PostGIS. CartoWeb puede ser configurado como un servicio Web SOAP[13] que permite tener los servidores Web y los servidores de datos de forma distribuida (Cartoweb, 2008). Este framework brinda a GeneSIG la posibilidad de trabajar con mapas entre otras funcionalidades, además de una estructura física de componentes que en principio es la misma que propone el Cartoweb. Esto permite la reutilización de dichos componentes en otros proyectos y la separación de la lógica del servidor (cartoserver), encargado del diálogo con el servidor de mapas y la provisión de servicios a un cliente (cartoclient).

1.6.4.3 GeneSIG

La plataforma de desarrollo de Sistemas de Información Geográfica "GeneSIG" debe su surgimiento y evolución al amplio desarrollo que han adquirido los SIG a nivel mundial, incrementando la efectividad y disminuyendo costos. Varias instituciones de diferentes países están inmersas en la búsqueda de herramientas de este tipo, para realizar nuevos sistemas o mejorar soluciones existentes. GeneSIG fue desarrollada por el centro de desarrollo GEySED[14], perteneciente a la Universidad de las Ciencias Informáticas, implementada con herramientas y tecnologías libres, cumpliendo con la necesidad actual de migración para lograr la independencia tecnológica. Su construcción está basada en estándares OpenGIS[15], incluyendo funcionalidades operativas en aplicaciones de este tipo.

GeneSIG está desarrollado sobre las siguientes tecnologías y herramientas: CartoWeb como framework PHP del lado del servidor y ExtJS del lado del cliente, implementado en los entornos integrados de desarrollo Zend Studio y Aptana, haciendo uso de la herramienta CASE[16] Visual Paradigm 3.4, sobre Sistemas Operativos Ubuntu 9.04 y Windows XP, MapServer[17] como servidor de mapas y Servidor Web Apache 2.5.6. Algunos de los componentes de la plataforma son la base de datos geoespacial, visor Web interactivo de acceso a los datos, servicio de catálogo y módulo de análisis. La utilización de GeneSIG para la construcción del sistema permite reducir el tiempo de desarrollo, debido a la cantidad de módulos ya implementados que realizan las funcionalidades más

[11] Empresa orientada a servicios e integración de aplicaciones de software libre para soluciones geoespaciales. Sociedad Anónima
[12] General Public License (Licencia Pública General).
[13] Simple Object Access Protocol (Protocolo Simple de Acceso a Objetos).
[14] Centro de Desarrollo Geoinformática y Señales Digitales.
[15] Estándar internacional orientado a Sistemas de Información Geográfica.
[16] Computer Aided Software Engineering (Ingeniería de Software Asistida por Computadora).
[17] Plataforma de código abierto para la publicación de datos espaciales.

comunes en un SIG. Es importante destacar la madurez de la plataforma además de la aceptación entre los desarrolladores que la utilizan, sin dejar de mencionar que sus módulos se pueden adaptar a diferentes negocios.

1.6.4.4 Servidor Web Apache

Apache es un servidor Web de código abierto para plataformas Unix, Microsoft Windows, Macintosh, potente y estable, sencillo para las tareas más comunes de mantenimiento. Es indiscutiblemente uno de los mayores logros del Software Libre. Este proyecto se desarrolla dentro del HTTP Server de la Apache Software Foundation. Es altamente configurable, su robustez y estabilidad lo han convertido en el servidor Web por excelencia. Es un servidor seguro, eficiente y extensible que proporciona servicios HTTP en sincronía con los estándares HTTP actuales (Apache, 2011). El servidor de mapas MapServer necesita un servidor Web Apache para su funcionamiento en plataformas Linux sobre la cual se desarrolla el SIG, por tal motivo se ha seleccionado su uso, además de poseer características como robustez, licencia de código abierto entre otras.

1.6.4.5 Servidor de Mapas MapServer

Un servidor de cartografía digital también conocido IMS[18], provee cartografías a través de la red tanto en modo vectorial como con imágenes rasterizadas. MapServer es un entorno de desarrollo de código abierto para la creación de SIG en la Web, con el fin de visualizar, consultar y analizar información geográfica a través de la red mediante la tecnología Internet Map Server (IMS). MapServer es uno de los servidores más rápidos bajo carga moderada, el SIG-MES será usado por el departamento de estadísticas de la misma institución, donde el número de usuarios no excederá de 5 personas. La selección de este servidor de mapas se basa en su rendimiento, además de ser tecnología libre se ajusta perfectamente a Sistemas de Información Geográfica Web.

1.6.4.6 NetBeans como IDE para PHP

NetBeans es un IDE[19] que posee un excelente completamiento de código. 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 (NetBeans, 2011). Permite desarrollar aplicaciones para diferentes entornos como escritorio y Web, utilizando como lenguaje Java, C/C++, Ruby on Rails, PHP, Groovy, Python, JavaScript entre

[18] Internet Map Server (Servidor de Mapas de Internet).
[19] Integrated development environment (Entorno Integrado de Desarrollo).

otros. Presenta mejoras para SOA[20] y UML[21]. Después de haber analizado varias herramientas de este tipo, e identificar características similares, como licencias, completamiento de código e integración con los lenguajes de programación empleados para la construcción del SIG, se decide su uso debido a la experiencia y dominio del equipo de desarrollo sobre este IDE. De esta forma se eleva la productividad y se garantiza la calidad del producto final.

1.6.4.7 Visual Paradigm como Herramienta Case

Las herramientas CASE son 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 permiten incrementar la productividad y el control de la calidad en cualquier proceso de elaboración de software, pues transforman la actividad de desarrollar software en un proceso automatizado. (Valle, 2009)

Visual Paradigm utiliza UML como lenguaje de modelado. Esta herramienta está pensada para usuarios interesados en construir sistemas de software fiables con el uso del paradigma orientado a objetos, incluyendo actividades entre las que se destacan: Ingeniería de software, análisis de sistemas y análisis de negocios. Emplea las últimas notaciones de UML, ingeniería inversa, generación de código, importación de Rational Rose, exportación e importación XML. Es tecnología libre y está disponible en varios idiomas, es fácil de instalar y actualizar. La utilización de la herramienta en cuestión ayudará a aumentar la productividad durante el desarrollo del SIG-MES, automatizando parte del proceso, además, la documentación será enriquecida con diagramas para lograr mejor entendimiento del funcionamiento del sistema por futuros desarrolladores.

1.6.5 Sistemas Gestores de Bases de Datos

1.6.5.1 PostgreSQL

PostgreSQL, originalmente se llamó Postgres, fue creado en la Universidad de California en Berkeley, por un profesor de Ciencia de la Computación llamado Michael Stonebraker. PostgreSQL es un potente gestor de base de datos relacional libre, bajo licencia BSD[22]. Cuenta con más de 15 años de desarrollo activo y arquitectura probada que ha ganado mucha reputación por su confidencialidad e integridad en los datos. (PostgreSQL, 2012). Su uso se debe a que este gestor está diseñado para

[20] Service Oriented Architecture (Arquitectura Orientada a Servicios).
[21] Unified Modeling Language (Lenguaje Unificado de Modelado).
[22] Berkeley Software Distribution (Distribución de software Berkeley).

ambientes de altos volúmenes de datos, debido a su estrategia de almacenamiento de filas llamada MVCC[23]. Al ser multiplataforma está disponible en casi cualquier sistema operativo. En términos de eficiencia y recursos, es capaz de ajustarse al número de procesadores y a la cantidad de memoria que posee el sistema de forma óptima, lo que posibilita atender un mayor número de peticiones concurrentemente. GeneSIG propone su uso para el desarrollo de sistemas con dicha plataforma.

1.6.5.2 PostGIS

PostGIS es un módulo que añade soporte de objetos geográficos a la base de datos objeto-relacional PostgreSQL, convirtiéndola en una base de datos espacial y así posibilitando su uso en Sistemas de Información Geográfica. Publicado bajo Licencia Pública General de GNU[24]. Su desarrollo y soporte está a cargo de la empresa canadiense Refraction Research (PostGIS, 2011). PostGIS es muy utilizado actualmente ya que ha demostrado estabilidad y eficiencia en relación con otros productos, demostrando superioridad sobre extensiones geográficas como la de MySQL, es similar a la versión geográfica del potente gestor de bases de datos Oracle. Este sistema ha sido certificado por el Open Geospatial Consortium (OGC) lo que garantiza la interoperabilidad con otros sistemas. PostGIS proporciona formas de almacenado que favorecen y facilitan su uso. Soporta estructura de datos para líneas, puntos y polígonos, las cuales son muy usadas en sistemas como el que se propone.

1.7 Metodologías de Desarrollo

Las metodologías de desarrollo de software son un conjunto de procedimientos, técnicas y ayudas a la documentación para el desarrollo de productos software. (Murcia, 2006), estas imponen un trabajo organizado con el objetivo de lograr mayor eficiencia. En ellas se van detallando los pasos a seguir en todas las actividades a realizar, para lograr un buen producto informático. Las metodologías de desarrollo indican las personas que deben participar en el desarrollo de dichas actividades y el rol que debe ocupar cada una de ellas.

Existen numerosas propuestas en cuanto a metodologías de desarrollos de software, fuertes, ágiles, pero el éxito de cada una consiste en seleccionar la adecuada y la que más se ajuste al entorno donde va a manifestarse. Es necesario entender los procesos de desarrollo y tener pleno dominio de los factores involucrados en estos, como tiempo, presupuesto entre otros.

[23] Multi-Version Concurrency Control.
[24] Licencia Pública General

1.7.1 RUP[25]

El Proceso Unificado de Desarrollo es un marco de trabajo genérico que se especializa en una gran variedad de sistemas de software, en distintas áreas de aplicación, tipos de organizaciones, niveles de aptitud y tamaños de proyecto. Es un proceso, que en su modelación define como principales elementos: trabajadores que conforman el (quién); actividades que representan el (cómo); artefactos que son los productos tangibles del proyecto que concretan el (qué); y el flujo de actividades que precisan la secuencia de actividades el (cuándo). Las características que se destacan en el ciclo de vida de RUP son:

- **Dirigido por casos de uso:** Orientan el proyecto a la importancia para el usuario y lo que este quiere. Los casos de uso representan los requisitos funcionales y fuerzan a pensar en términos de importancia para el usuario. Estos casos de uso no sólo son una herramienta para especificar los requisitos del sistema, también guían todo el desarrollo de software.
- **Centrado en la arquitectura:** Realiza la toma de decisiones que indica cómo tiene que ser construido el sistema y en qué orden.
- **Es iterativo e incremental:** Divide el desarrollo en mini proyectos donde los casos de uso y la arquitectura cumplen su objetivo de manera más depurada.

RUP está compuesto por nueve flujos de trabajo, los cuales serán detallados a continuación.

- **Modelamiento del negocio:** Este flujo se realiza para fomentar el entendimiento entre clientes y desarrolladores para idear en conjunto las necesidades del negocio que se quieren automatizar.
- **Requerimientos:** Se realiza entre desarrolladores y clientes para definir lo que el sistema debe hacer y cumplir.
- **Análisis y Diseño:** Se describe y define cómo se implementará el sistema, centrándose de forma general en la visión que se tiene de la arquitectura.
- **Implementación:** Se crea el producto, asegurando que cumpla con lo establecido por el cliente y los desarrolladores en los flujos iniciales.
- **Pruebas:** Se realizan las pruebas pertinentes de software y hardware al producto para asegurar la calidad de este al ser entregado.
- **Despliegue:** El producto se hace llegar a sus usuarios finales y junto con esto la instalación.

[25] Rational Unified Process

- **Administración de configuración y cambios:** Su objetivo es mantener la integridad de todos los artefactos que se crean en el proceso, así como su actualización.
- **Administración de proyectos:** El objetivo de este flujo es lograr un balance en la gestión de los objetivos, riesgos y restricciones para desarrollar un producto que cumpla con las restricciones solicitadas por los usuarios.
- **Ambiente:** Se facilita una guía para la configuración del ambiente apropiado para la instalación del proyecto.

La arquitectura en cuestión cuenta con cuatro fases, en la investigación se profundizará la fase de construcción donde el rol de implementador juega un papel fundamental.

- **Inicio:** En esta fase se establece la visión y el alcance del proyecto, además las partes interesadas deben realizar la estimación de tiempo y costo.
- **Elaboración:** En esta etapa se analiza el dominio del problema para establecer una arquitectura base sólida para el desarrollo exitoso del proyecto.
- **Construcción:** En esta etapa el objetivo es llegar a obtener la capacidad operacional inicial, así como desarrollar y probar el producto.
- **Transición:** El objetivo fundamental es llegar a obtener la liberación del proyecto.

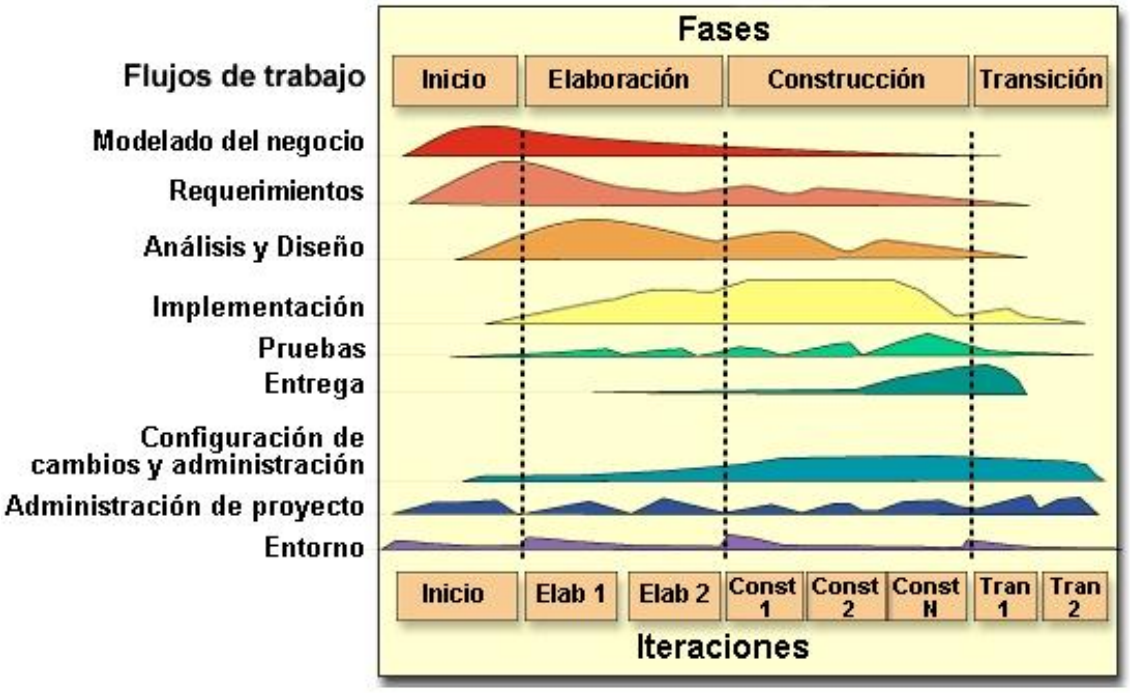

Figura 2 Flujos de trabajo y fases de RUP

La selección de una u otra metodología para el desarrollo de Software se debe a sus características fundamentales. Estas hacen diferir una metodología de otra y justifican su uso, ya que permiten ajustarse a las características del negocio que se desea informatizar. RUP brinda al equipo de desarrollo del SIG-MES una metodología robusta, donde una de sus principales características es que el proceso es iterativo incremental, de esta forma se logra mejorar el software en cada iteración. Es importante destacar la cantidad de artefactos bien documentados que genera, posibilitando mayor entendimiento entre los clientes y el equipo de desarrollo para futuras modificaciones y mejoras.

1.7.1.1 Lenguaje Unificado de Modelado (UML)

El lenguaje unificado de modelado es un lenguaje estándar de modelado para software, un lenguaje para la visualización, especificación, construcción y documentación de los artefactos de sistemas en los que el software juega un papel importante. Básicamente UML permite a los desarrolladores visualizar los resultados de su trabajo en esquemas o diagramas estandarizados. (Jacobson, 2000). Es un lenguaje visual orientado al modelado de sistemas. Facilita un vocabulario controlado por reglas y símbolos, con el fin de que todos los empleados de un proyecto obvien las ambigüedades y la dispersión conceptual. Dispone de un repertorio de unidades (clases, acciones, objetos, estados y casos de uso), estas características hacen de UML la herramienta necesaria para modelar el sistema que se propone en la presente investigación.

1.8 Conclusiones parciales

Luego de analizar en profundidad la situación problemática a resolver, para dar solución a las dificultades que presenta el MES en la actualidad y de abordar las principales características además de las consideraciones tecnológicas desde el punto de vista del rol implementador acerca de las herramientas y tecnologías que serán utilizadas en el desarrollo de la investigación, se logró seleccionar correctamente dichas tecnologías. Por tanto se concluye que el uso de estas será de vital importancia para lograr un desarrollo óptimo de la solución propuesta minimizando tiempo y costos. Teniendo como premisa fundamental que el sistema se desarrollará bajo tecnologías libres, fomentando las políticas de migración del país.

Capítulo 2

Análisis y construcción de la solución propuesta

2.1 Introducción.

En el presente capítulo, se realiza una valoración sobre el análisis del sistema elaborado por el analista. Se describen los patrones de diseño que se utilizaron en el desarrollo de la aplicación, en consonancia con la arquitectura definida. Se presenta el modelo de diseño e implementación del sistema con sus respectivos diagramas para un mejor entendimiento, además se mencionan los conceptos relacionados con las definiciones de estilo y estándares de código.

2.2 Valoración crítica sobre el análisis propuesto.

El análisis del sistema propuesto luego del estudio realizado en investigaciones anteriores resultó útil y sentó las bases para la construcción del Sistema de Información Geográfica, aunque el diagrama de casos de uso del sistema no presenta una correcta estructuración en cuanto a los actores que inicializan cada caso de uso, ya que un solo actor (Usuario) tiene acceso a todos los casos de uso del sistema. Se propone añadir un nuevo usuario (Administrador) que será el único actor que puede inicializar el caso de uso “Editar Referencia Geográfica”. A continuación se muestra la nueva propuesta del diagrama de casos de uso del sistema.

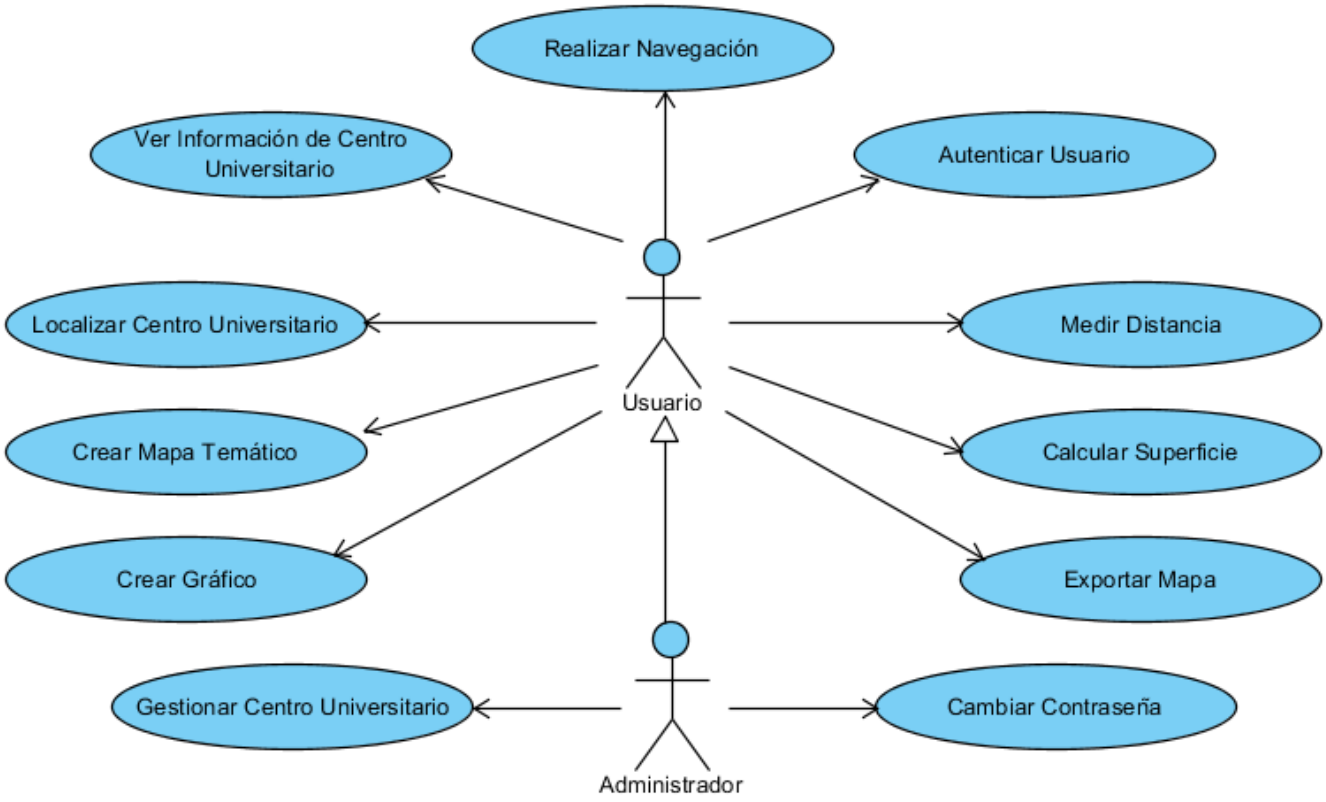

Figura 3 Diagrama de Casos de Uso del Sistema

2.3 Requisitos funcionales.

Los requisitos funcionales son características expresadas de las funcionalidades del sistema. Para una mejor comprensión los requisitos funcionales del SIG-MES han sido agrupados por módulos, los cuales se describen a continuación:

Módulo de Administración

RF1 Autenticar Usuarios: El sistema debe ser capaz de autenticar a los usuarios que tienen acceso al mapa.

RF2 Cambiar Contraseña: El sistema debe ser capaz de cambiar la contraseña de los usuarios que tienen acceso al mapa.

Módulo de navegación

RF3 Realizar Navegación: El sistema debe ser capaz de permitir realizar navegación en el mapa.

RF4 Acercar Mapa: El sistema debe ser capaz de acercar el mapa. Esta funcionalidad permite aumentar el tamaño del mapa y disminuye la escala, ubicando en el centro del mapa el punto en el que el usuario realizó la operación de Acercar.

RF5 Alejar Mapa: El sistema debe ser capaz de alejar el mapa. Esta funcionalidad disminuye el tamaño del mapa y aumenta la escala.

RF6 Recentrar Mapa: El sistema debe ser capaz de recentrar el mapa. Con este requisito se pretende que el usuario pueda recentrar una región previamente seleccionada al dar clic en el mapa, sin modificar su escala.

RF7 Ver Todo: El sistema debe ser capaz de permitir ver todo el mapa. Esta funcionalidad permite visualizar el mapa según la escala inicial de la aplicación.

RF8 Mover Mapa: El sistema debe ser capaz de mover el mapa. Con este requisito se pretende que el usuario pueda mover el mapa variando con el puntero del ratón la posición de la vista que se presenta.

RF9 Navegar a través del mapa de referencia: El sistema debe ser capaz de permitir al usuario navegar a través del mapa de referencia.

Módulo de consulta espacial

RF10 Ver Información Universidades: El sistema debe ser capaz de permitir ver información de las universidades. Esta funcionalidad tienes como objetivo que el usuario pueda consultar la información que está asociada a las universidades.

Módulo de localización

RF11 Localizar Centro Universitario: El sistema debe ser capaz de permitir al usuario localizar una universidad determinada en el mapa.

Módulo de edición

RF12 Gestionar Centro Universitario: El sistema debe ser capaz de permitir al usuario (Administrador) gestionar centros universitarios.

Módulo de análisis

RF13 Crear Mapas Temáticos: El sistema debe ser capaz de crear mapas temáticos a partir de los criterios introducidos por el usuario.

RF14 Crear Gráficos: El sistema debe ser capaz de crear gráficos a partir de la información brindada en el mapa.

RF15 Medir distancia: El sistema debe ser capaz de medir la distancia entre dos o más puntos en el mapa.

RF16 Calcular Superficie: El sistema debe ser capaz de calcular el área y perímetro de una región determinada en el mapa.

Módulo de impresión

RF17 Exportar Mapa: El sistema debe ser capaz de exportar el mapa.

2.4 Generalidades de la implementación.

2.4.1 Modelo de Implementación.

El modelo de implementación describe como se establece la estructura de los componentes y los elementos de implementación. Se basa en las responsabilidades asignadas a subsistemas y módulos. Los diagramas de componentes revelan cómo se divide el software, mostrando las dependencias entre sus partes. Por otro lado el diagrama de despliegue muestra la disposición física en a que se va a estacionar el sistema y las relaciones en la misma.

2.4.1.1 Diagrama de componentes.

La siguiente imagen muestra el diagrama de componentes correspondiente el caso de uso “Crear Mapa Temático”.

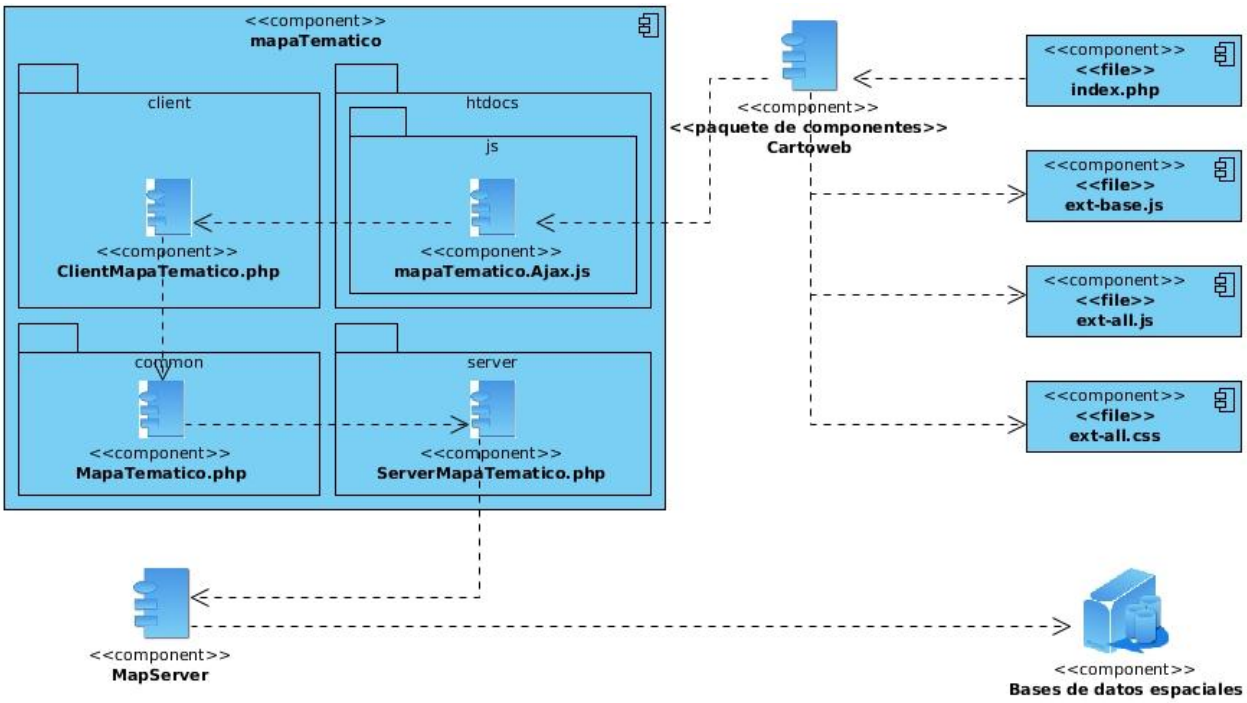

Figura 4 Diagrama de Componentes caso de uso "Crear Mapa Temático"

2.4.1.2 Modelo de Despliegue.

Los elementos de diseño al nivel del despliegue indican cómo se ubicará la funcionalidad y los subsistemas del entorno computacional físico que soportará al software (PRESSMAN, 2005).

El diagrama de despliegue muestra la disposición física de los nodos que componen el sistema y el reparto de los componentes sobre dichos nodos. Un nodo es un elemento físico que existe en tiempo de ejecución y representa un recurso computacional.

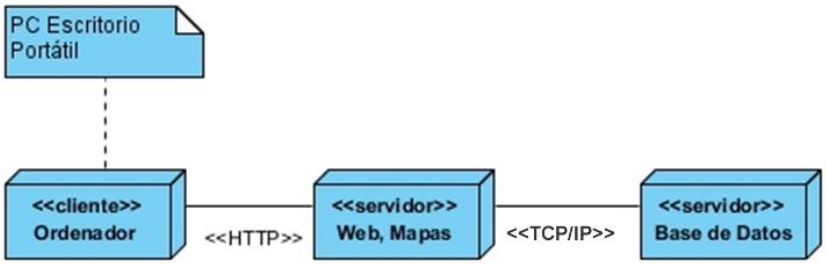

Figura 5 Diagrama de Despliegue: SIG-MES

El SIG-MES será utilizado desde un ordenador que puede ser una PC de escritorio o portátil. Su comunicación con el servidor Web y de Mapas se realiza mediante el protocolo de Transferencia de Hipertexto (HTTP), dichos servidores estarán conectados al servidor de Base de Datos mediante TCP/IP[26].

2.5 Estilos de programación.

El código de un programa lo leen muchas personas, bien para corregirlo, para ampliarlo o simplemente para evaluarlo. Para estas personas es fundamental que el código se encuentre bien redactado para que su significado sea claro y nítido. Sin embargo la legibilidad de un programa no depende sólo de las características del lenguaje sino que depende en gran medida del estilo de codificación en que está escrito. A diferencia de la sintaxis de un programa que son reglas estáticas que obligatoriamente hay que seguir, un estilo de programación está constituido por directrices que ayudan a obtener programas más legibles. No existen estilos de programación correctos o incorrectos, es aconsejable la adopción de un conjunto de normas para la escritura de programas. Estas normas, básicamente, se refieren a la forma en que se colocan las llaves, se indenta el código y como se ubican los paréntesis.

Existen varios criterios que describen un buen estilo de codificación, algunos de estos son:

1. **Nombres significativos:** Los nombres de funciones y variables definidos por el programador deben ser significativos y auto-explicativos con respecto a su función. Ejemplo: Si se debe designar una variable que almacenará la edad de una persona una forma correcta de hacerlo sería:
   $edad = $persona->getEdad(); y no $x = $a->getEdad();
2. **Indentación y espacios en el código:** Muestra las líneas que se encuentran subordinadas a otras líneas. Ejemplo:

```
if ($var == true)
{
        $cantidad++;
        $var = !$var;
}
```

3. **Documentar el código:** Las complicadas e inusuales secciones de código, así como las funciones deben ser comentadas para facilitar su comprensión.

[26] Transmission Control Protocol/Internet Protocol (Protocolo de Control de Transmisión/Protocolo de Internet).

Existen estilos de programación que por su fácil estructura y organización se han convertido en los más conocidos y utilizados por los desarrolladores pudiéndose mencionar dentro de ellos a los siguientes:

Estilo K&R: El estilo K&R es el más usado en el lenguaje C y PHP. Se trata de abrir la llave en la misma línea de declaración de la orden, indentando los siguientes pasos al mismo nivel que la llave y cerrando la llave en el mismo nivel que la declaración (CENDITEL, 2009). En este estilo la apertura no requiere una línea extra y la llave de finalización se alinea conceptualmente a la declaración que pertenece.

Ejemplo:

```
function Ejemplo($var) {
        if($var==1) {
           echo 1;
        } else {
           echo 0;
        }
}
```

Estilo Allman: El estilo Allman fue definido por Eric Allman. Se trata de crear una nueva línea para las llaves, e indentar el código debajo de ellas. La llave de cierre tiene el mismo indentado que la de inicio (CENDITEL, 2009). Se añade un salto de línea después del cierre de los paréntesis de los parámetros en las funciones además un salto de línea después un punto y coma, cuando termina la sentencia.

Ejemplo:

```
function Ejemplo($var)
{
        if($var==1)
        {
           echo 1;
        }
        else
        {
           echo 0;
        }
}
```

Estilo GNU: Parecido al estilo Allman, se trata de poner las llaves en una nueva línea. La diferencia es que las llaves deben estar indentadas por 2 espacios y el código dentro de ellas debe estarlo por 2 espacios más.

Ejemplo:

```
function Ejemplo($var)
 {
   if($var==1)
    {
      echo 1;
    }
   else
    {
      echo 0;
    }
 }
```

Para la implementación del SIG-MES se ha utilizado el estilo Allman, ya que mantiene un código limpio y claro, haciendo más fácil la indentificación de cada bloque. Además, al ser agradable visualmente es menos propenso a que los implementadores cometan errores.

2.6 Estándares de codificación.

Un estándar de codificación completo comprende todos los aspectos de la generación de código. Si bien los programadores deben implementar un estándar de forma prudente, éste debe tender siempre a lo práctico. Un código fuente completo debe reflejar un estilo armonioso, como si un único programador hubiera escrito todo el código de una sola vez. Al comenzar un proyecto de software, debe establecerse un estándar de codificación para asegurar que todos los programadores del proyecto trabajen de forma coordinada. Cuando el proyecto de software incorpore código fuente previo, o bien cuando realice el mantenimiento de un sistema de software creado anteriormente, el estándar de codificación debería establecer cómo operar con la base de código existente. (MSDN, 2012).

Usar técnicas de codificación sólidas y realizar buenas prácticas de programación con vistas a generar un código de alta calidad es de gran importancia para la eficacia del software y para obtener un buen rendimiento. Además, si se aplica de forma continuada un estándar de codificación bien definido, se utilizan técnicas de programación apropiadas, y, posteriormente, se efectúan revisiones del código de rutinas, caben muchas posibilidades de que un proyecto de software se convierta en un sistema fácil de comprender y de mantener. (MSDN, 2012).

Durante la implementación del SIG-MES se utilizará un estándar de codificación que pone en práctica una notación muy conocida por los implementadores, además de ser la utilizada en el proyecto

productivo “Aplicativos SIG” de la Universidad de las Ciencias Informáticas, donde se desarrollará la solución informática de la presente investigación.

Notación CamellCasing: Es la práctica de escribir frases o palabras compuestas en el que las palabras se unen sin espacios y se capitalizan. La notación consiste en escribir los identificadores con la primera letra de cada palabra en mayúscula y el resto en minúscula (CENDITEL, 2009).

Ejemplo:

```
$cantidadTuplasDevueltas=0;
```

2.7 Conclusiones parciales

Luego de haber realizado la valoración crítica del análisis propuesto y haber rectificado el diagrama de casos de uso del sistema, debido que no presentaba una correcta estructuración en cuanto a los actores que inicializan los casos de uso, permitirá mejorar el funcionamiento de la aplicación en cuanto a seguridad para los usuarios que accedan al SIG-MES. La realización del modelo de implementación, permitió mejor entendimiento de los componentes que conformarán el sistema. Por tanto se arriba a la conclusión que las bases están creadas para la implementación, además destacar el estándar de codificación que será utilizado, elemento indispensable a la hora de documentar y dar mantenimiento al sistema. Se puede afirmar que la construcción del SIG-MES será llevada a cabo de manera exitosa.

Capítulo 3

Validación de la solución propuesta

3.1 Introducción.

En este capítulo se pretende validar la solución propuesta, mediante la construcción de casos de pruebas de Caja Negra orientados a comprobar el funcionamiento de los casos de uso “Gestionar Centro Universitario” y “Crear gráfica”, sin llegar a probar directamente el código. El objetivo en este capítulo es la construcción de los casos de pruebas para las principales funcionalidades de dichos casos de uso, cumpliendo así con una de las tareas que plantea RUP para el rol de implementador.

3.2 Descripción general de las pruebas.

La prueba es el proceso de ejecución de un programa con la intención de descubrir un error. Las pruebas del software son un elemento crítico para la garantía de la calidad del software y representa una revisión final de las especificaciones, del diseño y la codificación. La creciente percepción del software como un elemento del sistema y la importancia de los costes asociados a un fallo del propio sistema, están motivando la creación de pruebas minuciosas y bien detalladas (PRESSMAN 2005).

Los errores dentro del proceso de desarrollo del software pueden venir dados desde su creación durante las fases de inicio y análisis hasta su explotación durante el despliegue. La identificación errónea de requisitos o un mal diseño de clases pueden provocar faltas graves en el buen funcionamiento de una aplicación. De ahí que se realicen a las aplicaciones de software numerosas pruebas con el objetivo de descubrir fallas no detectadas hasta ese momento.

Las pruebas no aseguran que el software presenta ausencia de defectos pero si demuestra que el software puede tener imperfecciones. Existen diferentes tipos de pruebas que se le pueden aplicar al software para identificar fallos en la aplicación, como son:

- **Pruebas de Integración**: Comprueban la compatibilidad y funcionalidad de los interfaces entre los distintos elementos que componen un sistema.
- **Pruebas de Verificación:** Se comprueba el cumplimiento de las especificaciones del diseño.

- **Pruebas de Validación:** Son realizadas sobre un software completamente integrado para evaluar el cumplimiento con los requisitos especificados.
- **Prueba de Caja Blanca:** Se basa en el diseño de casos de prueba que usen la estructura de control del diseño procedimental para derivarlos, en otras palabras se analiza la estructura lógica del programa.
- **Prueba de Caja Negra:** Se centra principalmente en los requisitos funcionales del software reflejados en su interfaz sin tener en cuenta el funcionamiento interno de la aplicación, no considera la codificación dentro de los parámetros a evaluar. Se basa en que las entradas sean aceptadas de forma adecuada y se reciba una salida correcta demostrando que cada función es completamente operativa.
- **Prueba de Aceptación:** Es la prueba final basada en las especificaciones del usuario. Su objetivo principal es demostrarle el cumplimiento del requisito de software al usuario. Puede estar asociado tanto a requisitos funcionales como no funcionales y cada requisito puede tener una o más pruebas de aceptación asociada.

Los diferentes métodos para la realización de las pruebas se analizaron detalladamente, debido a su importancia en la detección de posibles errores. Se decide usar las pruebas de Caja Negra, ya que la plataforma GeneSIG es un producto informático liberado, con varios años de explotación, además ha transitado por varios procesos de intensiva revisión.

3.3 Pruebas de Caja Negra

Para desarrollar la prueba de Caja Negra existen varias técnicas entre ellas:

- **Técnica de la Partición de Equivalencia:** Divide el campo de entrada en clases de datos que tienden a ejercitar determinadas funciones del software.
- **Técnica del Análisis de Valores Límites**: Prueba la habilidad del programa para manejar datos que se encuentran en los límites aceptables.
- **Técnica de Grafos de Causa-Efecto:** Permite al encargado de la prueba validar complejos conjuntos de acciones y condiciones.

Muchos autores consideran que estas pruebas permiten encontrar: (Pressman, 2005)

- Funciones incorrectas o ausentes.
- Errores de interfaz.

- Errores en estructuras de datos o en accesos a las Bases de Datos externas.
- Errores de rendimiento.
- Errores de inicialización y terminación.

“El diseño de casos de prueba para la partición equivalente se basa en una evaluación de las clases de equivalencia para una condición de entrada…una clase de equivalencia representa un conjunto de estados validos o no válidos para condiciones de entrada. Típicamente, una condición de entrada es un valor numérico específico, un rango de valores, un conjunto de valores relacionados o una condición lógica.” (Pressman, 2000)

La técnica de la Partición de Equivalencia es una de las más efectivas pues permite examinar los valores válidos e inválidos de las entradas existentes en el sistema, descubre de forma inmediata una clase de errores que, de otro modo, requerirían la ejecución de muchos casos antes de detectar el error genérico.

3.4 Diseño de las pruebas para validar la solución propuesta

A continuación se muestra el caso prueba para la sección “Insertar centro universitario” del caso de uso “Gestionar Centro Universitario” utilizando la técnica de Partición de Equivalencia.

Caso de Uso: Gestionar Centro Universitario.

Descripción general del caso de uso: El caso de uso inicia cuando el administrador necesita gestionar un centro universitario, insertando, modificando o eliminando determinado centro y termina cuando se visualiza en el mapa los cambios realizados.

Condiciones de Ejecución: Solo puede realizar esta acción el administrador y debe para ello dar clic sobre una provincia del mapa.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
SC 1: “Insertar centro universitario”.	EC 1.1: “Insertar centro universitario” con éxito.	El administrador inserta un centro universitario a partir de sus datos, dicho centro se añade al sistema y se muestra en el mapa un punto que lo representa.	1. Interfaz Principal. 2. Barra de herramientas. 3. Gestionar centro universitario. 4. Insertar centro universitario. 5. Aceptar.
	EC 1.2: “Insertar centro universitario” con campos vacíos.	Si el administrador no llena los campos correspondientes, el sistema muestra un mensaje de error diciendo “Debe llenar todos los campos correctamente”.	1. Interfaz Principal. 2. Barra de herramientas. 3. Gestionar centro universitario. 4. Insertar centro universitario.

	EC 1.3: Cerrar la opción "Insertar centro universitario".	Al seleccionar el botón "Cancelar" el sistema cierra la ventana "Insertar centro universitario" sin guardar cambios y muestra la interfaz "Gestionar centro universitario".	1. Interfaz Principal. 2. Barra de herramientas. 3. Gestionar centro universitario. 4. Insertar centro universitario.

Tabla 1 Sección a probar en Insertar centro universitario

Descripción de las variables.

No	Nombre del campo	Clasificación	Requerido	Descripción
1	Nombre	Campo de texto	Si	Especifica el nombre del centro universitario.
2	Cantidad de masters	Campo de texto	Si	Especifica la cantidad de masters que hay en el centro universitario.
3	Cantidad de doctores	Campo de texto	Si	Especifica la cantidad de doctores que hay en el centro universitario.
4	Cantidad de instructores	Campo de texto	Si	Especifica la cantidad de instructores que hay en el centro universitario.
5	Primer año	Campo de texto	Si	Especifica la cantidad de estudiantes de primer año que hay en el centro universitario.

6	Segundo año	Campo de texto	Si	Especifica la cantidad de estudiantes de segundo año que hay en el centro universitario.
7	Tercer año	Campo de texto	Si	Especifica la cantidad de estudiantes de tercer año que hay en el centro universitario.
8	Cuarto año	Campo de texto	Si	Especifica la cantidad de estudiantes de cuarto año que hay en el centro universitario.
9	Quinto año	Campo de texto	Si	Especifica la cantidad de estudiantes de quinto año que hay en el centro universitario.
10	Carreras	Lista de texto	Si	Especifica la cantidad de carreras que se estudian en el centro universitario.
11	Descripción	Campo de texto	No	Especifica la descripción del centro universitario.

Tabla 2 Descripción de las variables en Insertar centro universitario

Matriz de datos (SC 1. Insertar centro universitario)

Escenario	EC 1.1: "Insertar centro universitario" con éxito.	EC 1.3: "Insertar centro universitario" con campos vacíos.	EC 1.4: Cerrar la opción "Insertar centro universitario".
Variable 1 Nombre	Universidad de La Habana	I:"Vacío"	N/A

Variable 2 **Cantidad de masters**	100	85	N/A
Variable 3 **Cantidad de doctores**	50	I:“Vacío”	N/A
Variable 4 **Cantidad de instructores**	350	400	N/A
Variable 5 **Primer año**	750	I:“Vacío”	N/A
Variable 6 **Segundo año**	580	500	N/A
Variable 7 **Tercer año**	600	550	N/A
Variable 8 **Cuarto año**	650	I:“Vacío”	N/A
Variable 9 **Quinto año**	595	400	N/A
Variable 10 **Carreras**	35	I:“Vacío”	N/A

Variable 11 **Descripción**	La Universidad de La Habana abrió su primer curso académico el 30 de noviembre de 1952.	I:"Vacío"	N/A
Respuesta del Sistema	El sistema muestra una interfaz donde se deben situar los datos, una vez insertados el sistema realiza las verificaciones pertinentes, y de esta forma queda insertado el centro universitario.	El sistema muestra un mensaje de error diciendo que debe llenar los datos correspondientes.	Al seleccionar el botón "Cancelar" el sistema cerrará la ventana.
Resultado de la Prueba	Satisfactorio.	Satisfactorio.	Satisfactorio.

Tabla 3 Matriz de datos en Insertar centro universitario

A continuación se muestran los casos de prueba para el caso de uso "Crear Gráfico" utilizando la técnica de Partición de Equivalencia.

Caso de Uso: Crear Gráfico.

Descripción general del caso de uso: El caso de uso inicia cuando el usuario necesita graficar por provincias o por centros universitarios, termina cuando se visualiza el tipo de gráfico escogido por el usuario.

Condiciones de Ejecución: El usuario debe estar autenticado y debe existir en el sistema información de centros universitarios.

Secciones a probar en el caso de uso

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
SC 1: "Crear gráfico por centro universitario".	EC 1.1: "Crear gráfico por centro universitario" con éxito.	El usuario selecciona los criterios por los cuales desea graficar, el sistema muestra la gráfica según su selección.	1. Interfaz Principal. 2. Barra de herramientas. 3. Graficar. 4. Centros universitarios. 5. Aceptar.
	EC 1.2: "Crear gráfico por centro universitario" con campos vacíos.	El usuario no introduce los campos correspondientes, el sistema muestra un mensaje de error diciendo "Debe llenar los campos correctamente".	1. Interfaz Principal. 2. Barra de herramientas. 3. Graficar. 4. Centros universitarios. 5. Aceptar.

Tabla 4 Sección a probar en Crear gráfico por centro universitario

Descripción de las variables.

No	Nombre del campo	Clasificación	Requerido	Descripción
1	Universidades	Campo de selección	Si	Especifica el nombre del centro universitario por el que se va a graficar.

2	Criterios	Campo de selección	Si	Especifica los elementos a tener en cuenta para realizar la gráfica.
3	Tipo de gráfico	Campo de selección	No	Especifica el tipo de gráfico por el que se desea graficar.

Tabla 5 Descripción de las variables en Crear gráfico por centro universitario

Matriz de datos (SC 1. Crear gráfico por centro universitario)

Escenario	Variable 1 Universidades	Variable 2 Criterios	Variable 3 Tipo de gráfico	Respuesta del Sistema	Resultado de la Prueba
EC 1.1: "Crear gráfico por centro universitario" con éxito.	Universidad de la Habana	Matrícula	Pastel	El sistema grafica por matrícula a la Universidad de la Habana en un gráfico de pastel.	Satisfactorio.
EC 1.2: "Crear gráfico por centro universitario" con campos vacíos.	Universidad de Matanzas	I:"Vacío"	Barras	El sistema muestra un mensaje de error diciendo que se deben llenar los campos correctamente.	Satisfactorio.

Tabla 6 Matriz de datos Crear gráfico por centro universitario

Nombre de la	Escenarios de la	Descripción de la funcionalidad	Flujo Central

sección	sección		
SC 2: "Crear gráfico por provincia".	EC 2.1: "Crear gráfico por provincia" con éxito.	El usuario selecciona la opción "Provincias", además de escoger el criterio para graficar. El sistema muestra la gráfica según los datos introducidos.	1. Interfaz Principal. 2. Barra de herramientas. 3. Graficar. 4. Provincias. 5. Aceptar.
	EC 2.2: "Crear gráfico por provincia" con campos vacíos.	El usuario no introduce los campos correspondientes, el sistema muestra un mensaje de error diciendo "Debe llenar los campos correctamente".	1. Interfaz Principal. 2. Barra de herramientas. 3. Graficar. 4. Provincias. 5. Aceptar.

Tabla 7 Sección a probar en Crear gráfico por provincia

Descripción de las variables.

No	Nombre del campo	Clasificación	Requerido	Descripción
1	Provincias	Campo de selección	Si	Especifica la provincia que se va a graficar.
2	Criterios	Campo de selección	Si	Especifica los elementos a tener en cuenta para realizar la gráfica.
3	Tipo de gráfico	Campo de selección	No	Especifica el tipo de gráfico por el que se desea graficar.

Tabla 8 Descripción de las variables en Crear gráfico por provincia

Matriz de datos (SC 2. Crear gráfico por provincia)

Escenario	Variable 1 Provincias	Variable 2 Criterios	Variable 3 Tipo de gráfico	Respuesta del Sistema	Resultado de la Prueba
EC 2.1: "Crear gráfico por provincia" con éxito.	Villa Clara	Cantidad de profesionales	Barras	El sistema grafica por cantidad de profesionales a la provincia de Villa Clara en un gráfico de barras.	Satisfactorio.
EC 2.2: "Crear gráfico por centro universitario" con campos vacíos.	I:"Vacío"	Matrícula	Pastel	El sistema muestra un mensaje de error diciendo que se deben llenar los campos correctamente.	Satisfactorio.

Tabla 9 Matriz de datos Crear gráfico por provincia

3.5 Conclusiones parciales

Los diseños de casos de pruebas presentados validaron algunas de las funcionalidades y garantizaron la calidad del Sistema de Información Geográfica para el MES. Luego de aplicar las pruebas de Caja Negra al software, las cuales se basan principalmente en el cumplimiento de los requisitos funcionales reflejados en la interfaz del sistema, aplicando la técnica de equivalencia, se detectaron dos no conformidades las que fueron corregidas inmediatamente. Debido a que este tipo de pruebas no es

muy complejo, brindan cumplimiento a algunos de los requisitos funcionales más importantes señalados por el analista. Las pruebas realizadas confirmaron la factibilidad de la solución propuesta.

Conclusiones

Una vez concluido el proceso investigativo y desarrollado el Sistema de Información Geográfica para el MES, se resaltan una serie de conclusiones las cuales son expuestas a continuación:

1. Las herramientas y tecnologías utilizadas en el desarrollo de la aplicación obedecen a criterios de selección de tecnologías libres y multiplataforma, adecuándose a las políticas que impulsa la universidad y el país.

2. Todos los requisitos funcionales previstos fueron implementados satisfactoriamente teniendo en cuenta las restricciones no funcionales especificadas a lo largo de la investigación.

3. La aplicación desarrollada es fácil de utilizar, intuitivo y con un diseño sencillo.

4. El diseño de las pruebas de Caja Negra permitió validar el cumplimiento de los requisitos funcionales y comprobar que el sistema efectúa las acciones que debe realizar en el momento esperado.

5. La puesta en explotación de la aplicación desarrollada proporciona mayor calidad y rapidez, en el proceso de investigación y análisis de los datos manipulados en el MES, eliminando errores que pueden producirse de forma manual y contribuyendo a mejores resultados para la toma de decisiones.

6. Se logró documentar todo el proceso de desarrollo del sistema para posteriores estudios y modificaciones.

Recomendaciones

En correspondencia con los resultados obtenidos y la experiencia acumulada a lo largo de todo el proceso investigativo se recomienda lo siguiente:

1. Establecer un período de capacitación para los usuarios del Ministerio de Educación Superior que trabajarán con el Sistema de Información Geográfica, con el objetivo de prepararlos para un buen desarrollo en sus actividades y funcionamiento con la aplicación.
2. Continuar el ciclo de desarrollo de la herramienta, con la identificación de nuevos requisitos funcionales, en aras de ampliar las posibilidades de trabajo del sistema desarrollado.
3. Socializar el resultado de esta investigación en algún espacio de la universidad para que sirva de consulta a investigaciones afines o similares.

Referencias Bibliográficas

Casanova, Josep. 2004. Desarrollo web. [Online] 2004. [Cited: Enero 7, 2012.] http://www.desarrolloweb.com/articulos/arquitectura-cliente-servidor.html.

CSS W3C, Consortium World Wide Web. World Wide Web Consortium. *CSS.* [Online] [Cited: 10 1, 2011.] http://www.w3.org/Style/CSS.

Fung, Ing. Luis Lamela. 2010. Mapping Interactivo. Revista Internacional de Ciencias de la Tierra. [Online] Septiembre 2010. [Cited: 10 1, 2011.] http://www.mappinginteractivo.com/plantilla.asp?id_articulo=1518.

Garlan, David. 1994. *An Introduction to Software Architecture.* 1994.

Gracia, J. 2005. Patrones de diseño. *Diseño de software Orientado a Objetos.* [Online] 2005. http://www.ingenierosoftware.com/analisisydiseno/patrones-diseno.php.

HTML W3C, Consortium World Wide Web. World Wide Web Consortium. [Online] [Cited: 10 1, 2011.] http://www.w3.org/TR/xhtml1/#xhtml.

Ivar Jacobson, Grady Booch, James Rumbaugh. *El proceso unificado de desarrollo de software.*

Jacobson, Ivar. 2000. *Proceso Unificado de Desarrollo de Software.* Madrid : s.n., 2000. ISBN:9788478290369.

Lanzillotta, Analía. 2005. Master Magazine. [Online] Febrero 2005. [Cited: 10 1, 2011.] http://www.mastermagazine.info/termino/5560.php.

Links Global, Services. Links Global Services. [Online] [Cited: 11 4, 2011.] http://www.lgs.com.ve/pres/PresentacionES_PSQL.pdf.

MINREX. Ministerio de Relaciones Exteriores de Cuba. [Online] [Cited: 10 1, 2011.] http://www.cubaminrex.cu/Sociedad_Informacion/Informacion_Gral.htm.

Morero, Francisco. 1999-2000. *Introduccion a la OOP.* s.l. : Grupo EIDOS, 1999-2000.

MSDN. MSDN. [Online] http://msdn.microsoft.com/es-es/library/aa291591%28VS.71%29.aspx.

Murcia, Universidad de. [Online] [Cited: Enero 21, 2012.] http://www.um.es/docencia/barzana/IAGP/Iagp2.html.

PHP. PHP: Hypertext Preprocessor. [Online] [Cited: 10 1, 2011.] http://www.php.net/manual/en/faq.general.php.

PostgreSQL. PostgreSQL. [Online] [Cited: 10 4, 2011.] http://www.postgresql.org/about.

Potencier, Fabien. *Symfony 1.1, la guía definitiva.*

Pressman Roger S Ingeniería de Software un Enfoque Práctico [Libro]. - 2000. - Vol. 6ta Edición.

Pressman, Roger S. 2005. *Ingeniería de Software. Un enfoque práctico.* 2005.

Reynoso. 2004. *Introducción a la Arquitectura de Software.* 2004.

Rico, José Lorenzo Herrero. 2000. Cartografía digital y Espeleología. [Online] Marzo 2000. [Cited: 10 1, 2011.] http://www.tic.udc.es/~nino/blog/documentos/cartografia-digital.pdf.

Rolando Alfredo Hernández León, Sayda Coello González. 2002. *El Paradigma Cuantitativo de la Investigación.* Ciudad de la Habana : EDUNIV Editorial Universitaria, 2002. ISBN: 959-16-0343-6.

San Marcos, Universidad Nacional Mayor. Portal del sistema de bibliotecas de la UNMSM. [Online] [Cited: 10 20, 2011.] http://sisbib.unmsm.edu.pe/bibVirtual/Publicaciones/indata/v04_n1/lenguajes.htm.

Silva, José Luis Batista. 2005. Aplicación de Sistemas de Información Geográfica en Cuba. *Mapping Interactivo. Revista Internacional de Ciencias de la Tierra.* [Online] 2005. [Cited: 10 1, 2011.] http://www.mappinginteractivo.com/plantilla-ante.asp?id_articulo=1051.

Thompson, Ivan. 2008. Definición de Información. [Online] 10 2008. [Cited: 10 1, 2011.] http://www.promonegocios.net/mercadotecnia/definicion-informacion.html.

Valle, Dianelys del. 2009. *Desarrollo del Portal WAP para la plataforma de gestión de contenidos Gina.* Habana : s.n., 2009.

Bibliografías Consultadas

Ávila Bárzaga, Olaidis. 2011. *SIG-MES: Análisis del sistema para la representación geográfica de los Centros Universitarios del Ministerio de Educación Superior Cubana.*

Garlan, David. 1994. *An Introduction to Software Architecture.* 1994.

Jacobson, Ivar. 2000. *Proceso Unificado de Desarrollo de Software.* Madrid : s.n., 2000. ISBN:9788478290369.

Larman, Craig. 1999. *UML y Patrones. Introducción al análisis y diseño orientado a objetos.* Primera edición. Prentice Hall, México : s.n., 1999.

Morero, Francisco. 1999-2000. *Introduccion a la OOP.* s.l. : Grupo EIDOS, 1999-2000.

Orallo, E. H. 2002. *El Lenguaje Unificado de Modelado (UML), Paraninfo Thomson Learning.* 2002.

PHP. PHP: Hypertext Preprocessor. [Online] [Cited: 10 1, 2011.] http://www.php.net/manual/en/faq.general.php.

Pockin. 2008. *Modelado de Sistemas con UML.* 2008.

Potencier, Fabien. *Symfony 1.1, la guía definitiva.*

Pressman Roger S. Ingeniería de Software un Enfoque Práctico [Libro]. - 2000. - Vol. 6ta Edición.

Pressman, Roger S. 2005. *Ingeniería de Software. Un enfoque práctico.* 2005.

Reynoso. 2004. *Introducción a la Arquitectura de Software.* 2004.

Rolando Alfredo Hernández León, Sayda Coello González. 2002. *El Paradigma Cuantitativo de la Investigación.* Ciudad de la Habana : EDUNIV Editorial Universitaria, 2002. ISBN: 959-16-0343-6.

Anexos

Anexo 1: Diagramas de componentes.

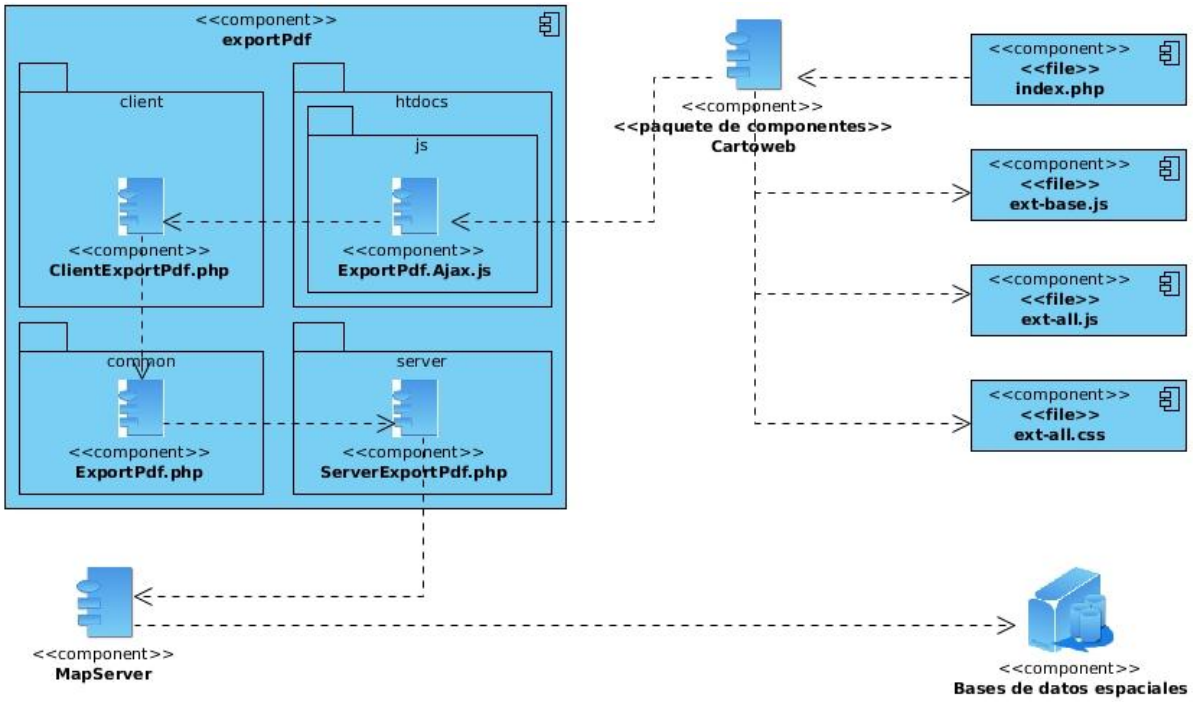

Diagrama de componentes: CU Exportar Mapa

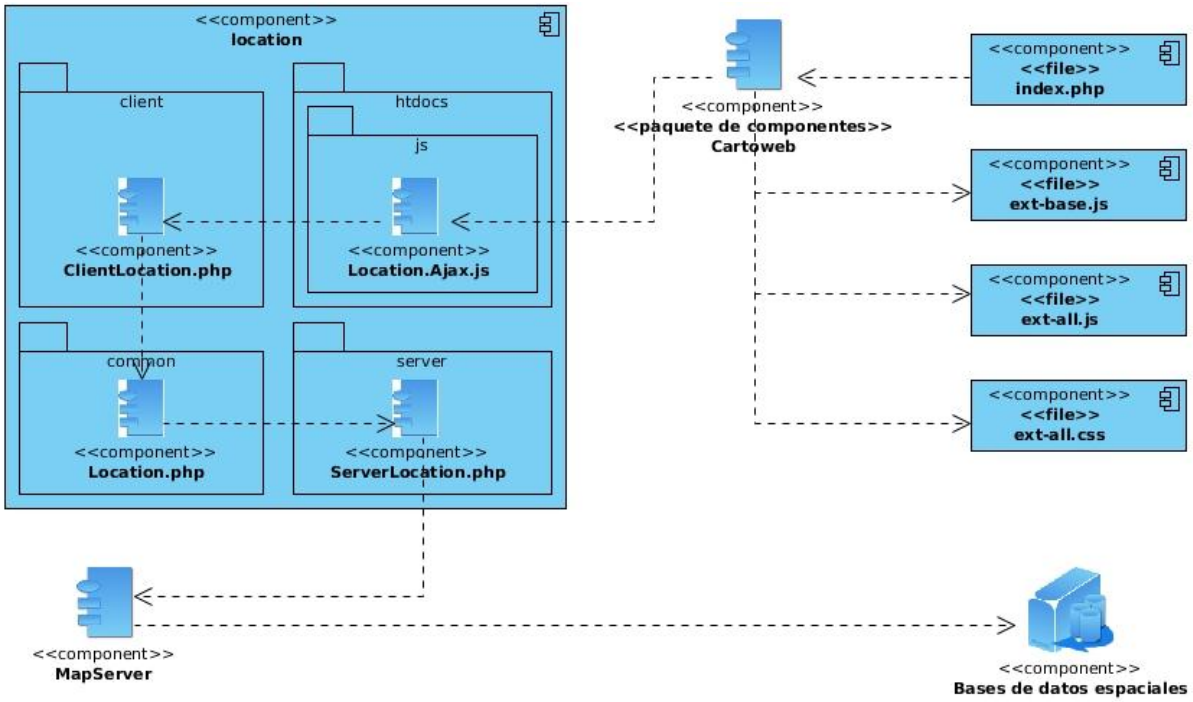

Diagrama de componentes: CU Realizar Navegación

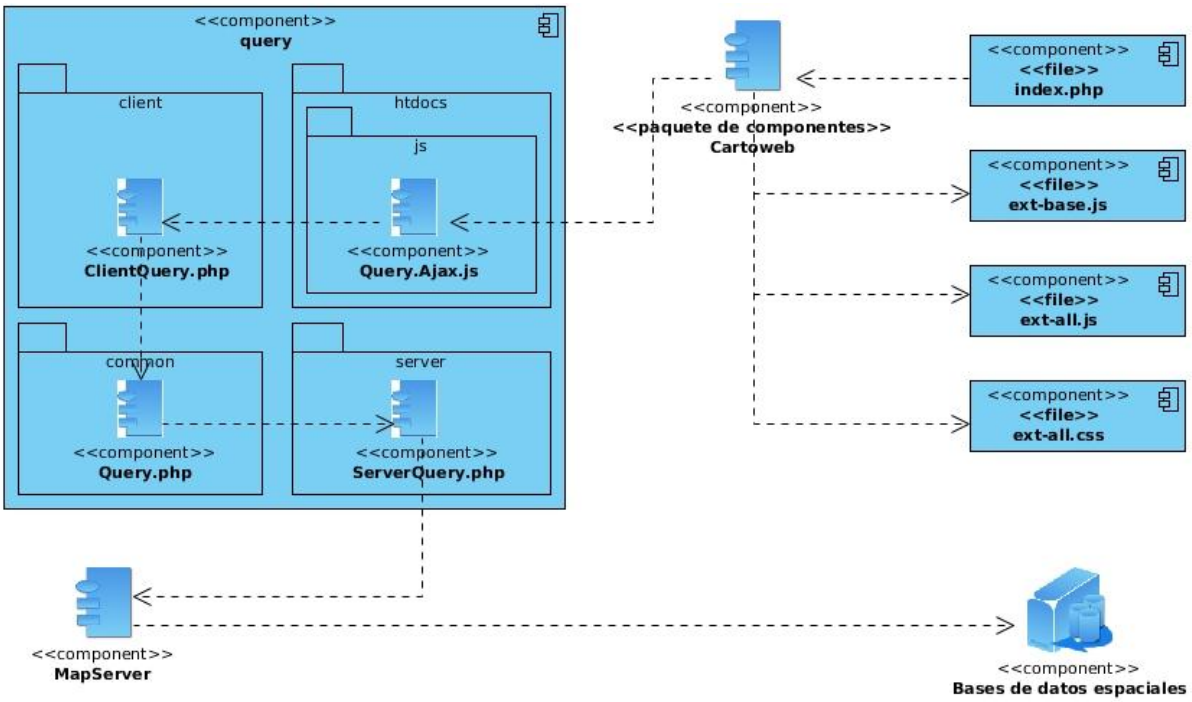

Diagrama de componentes: CU Ver Información de Centro Universitario

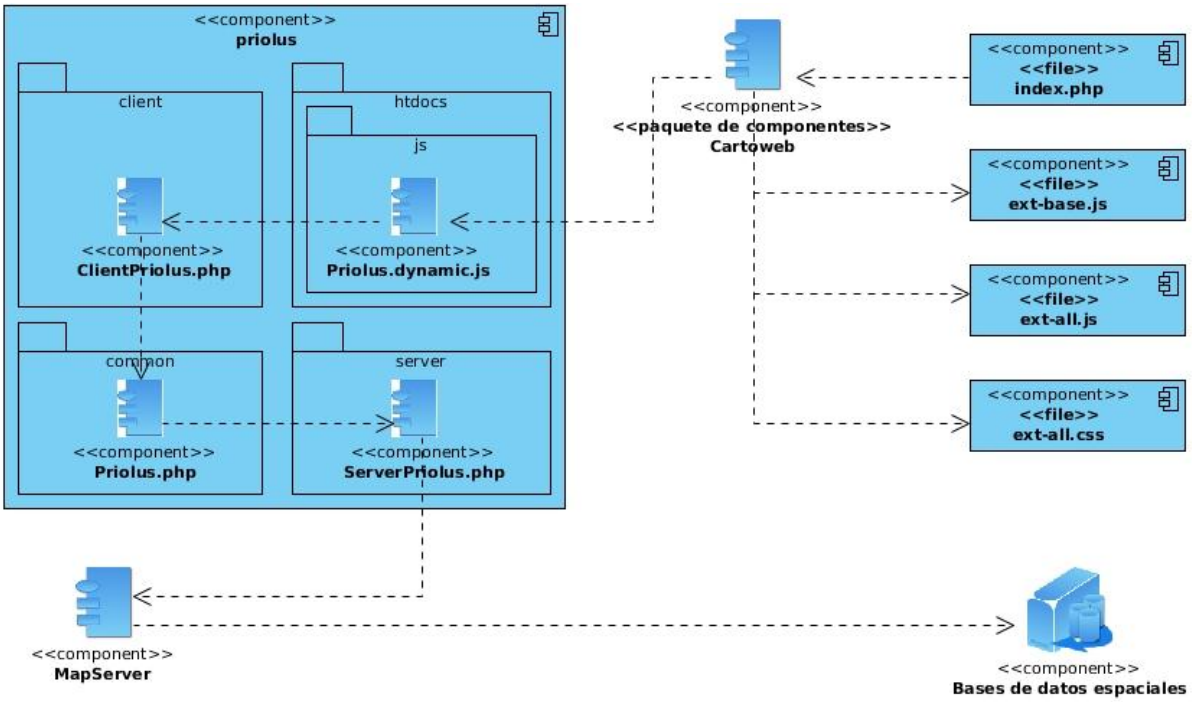

Diagrama de componentes: CU Gestionar Centro Universitario

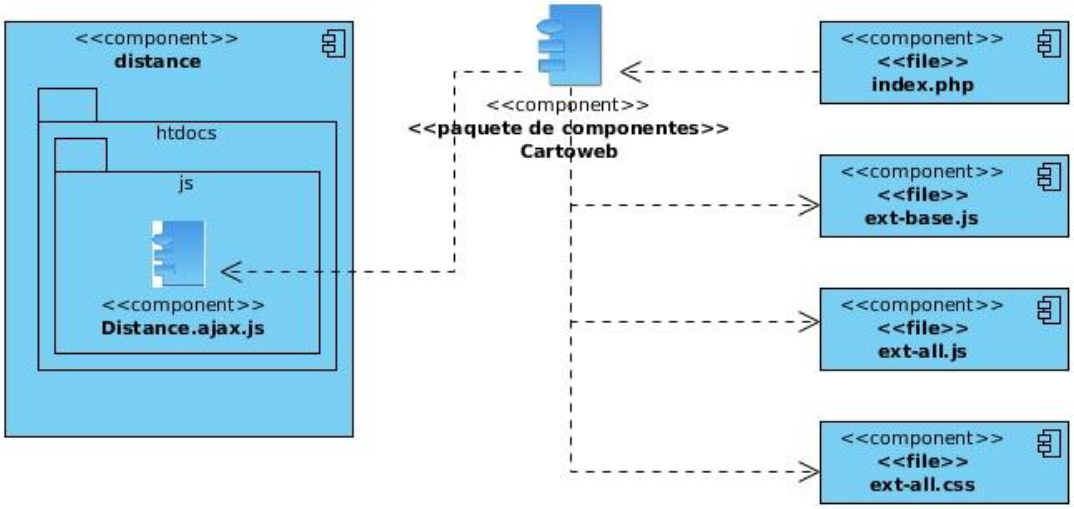

Diagrama de componentes: CU Medir Distancia

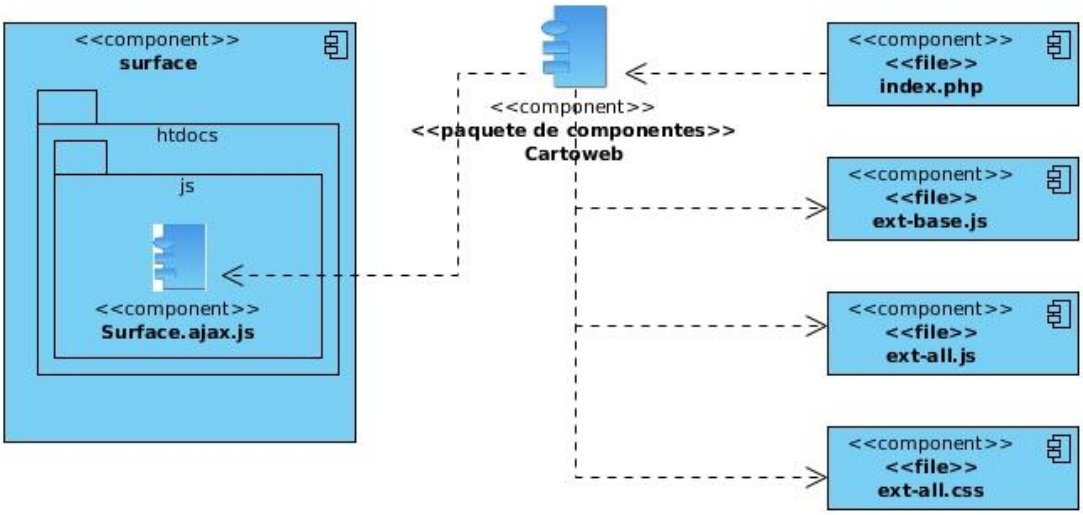

Diagrama de componentes: CU Calcular Superficie

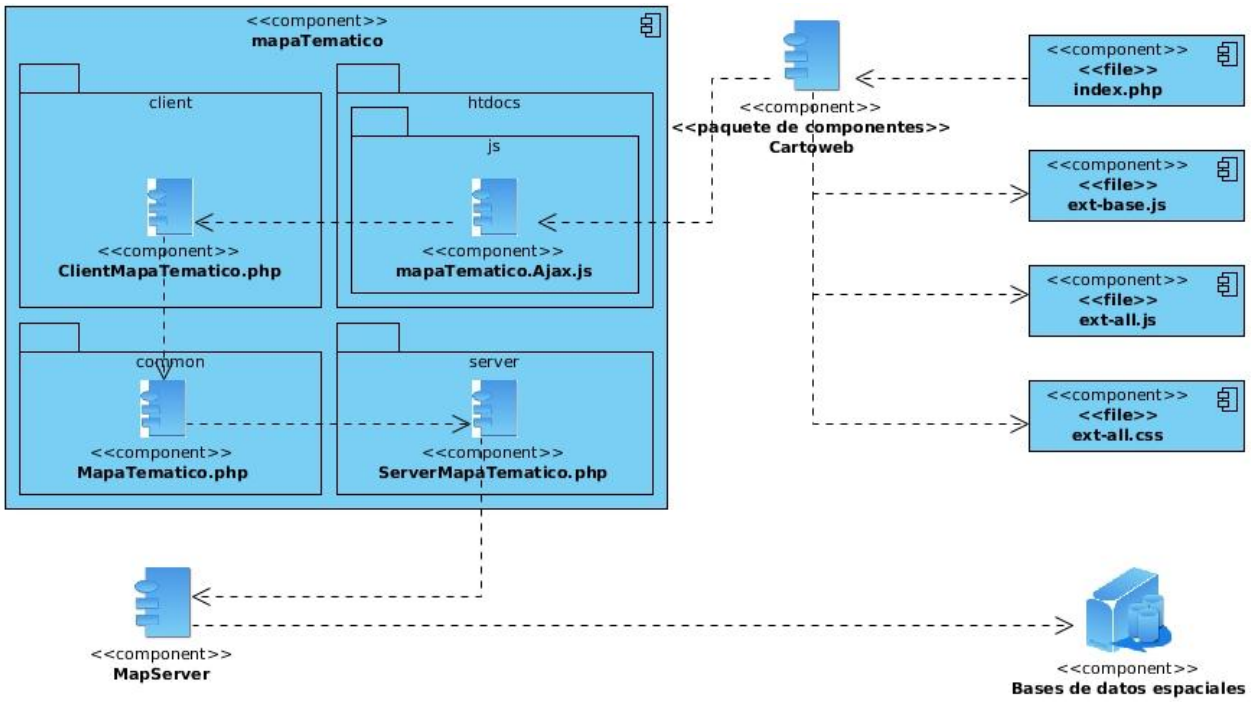

Diagrama de componentes: CU Crear Mapa Temático

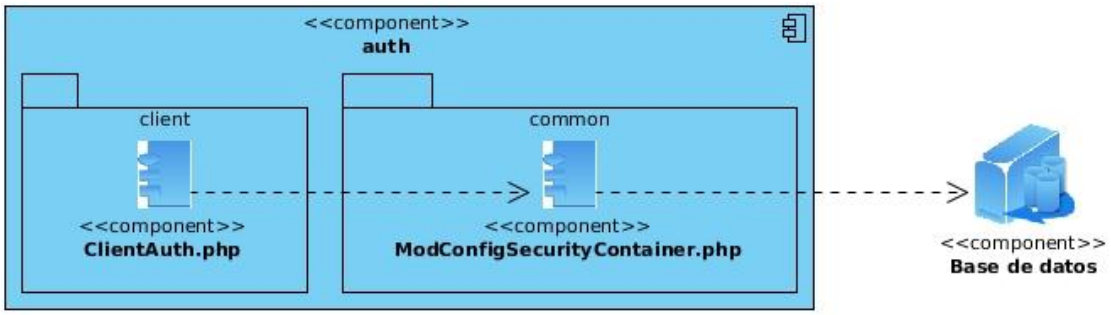

Diagrama de componentes: CU Autenticar Usuario

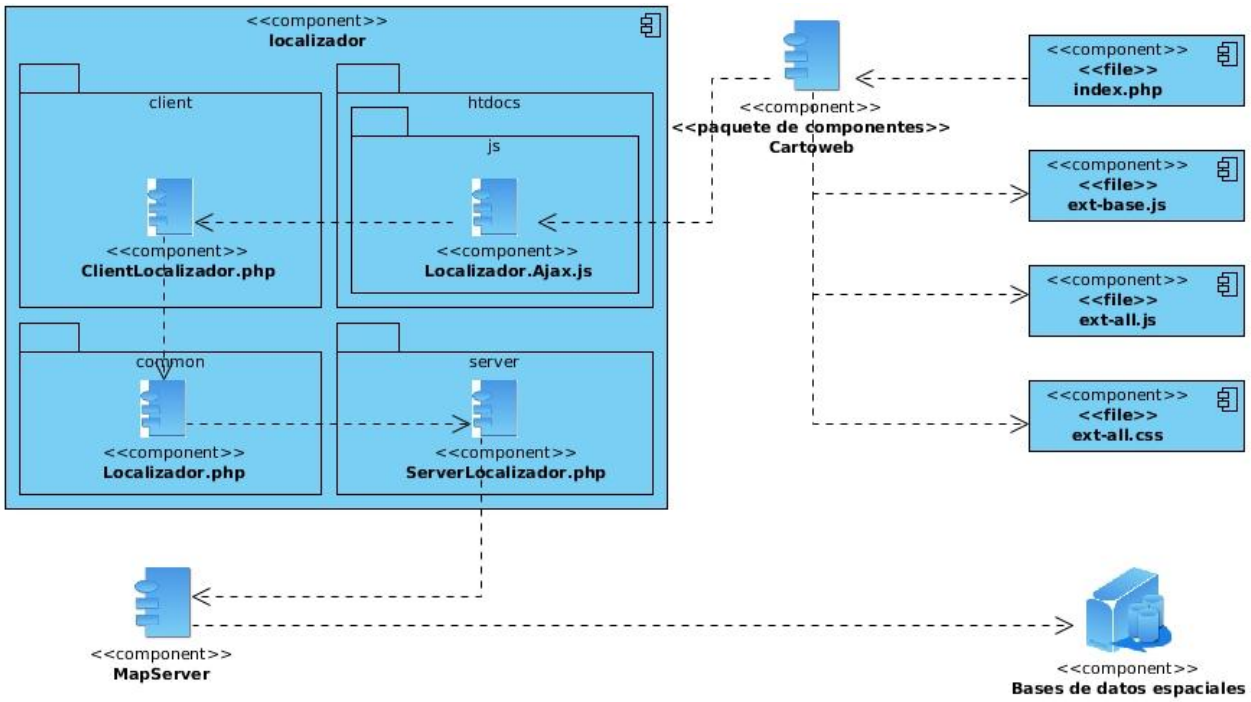

Diagrama de componentes: CU Localizar Centro Universitario

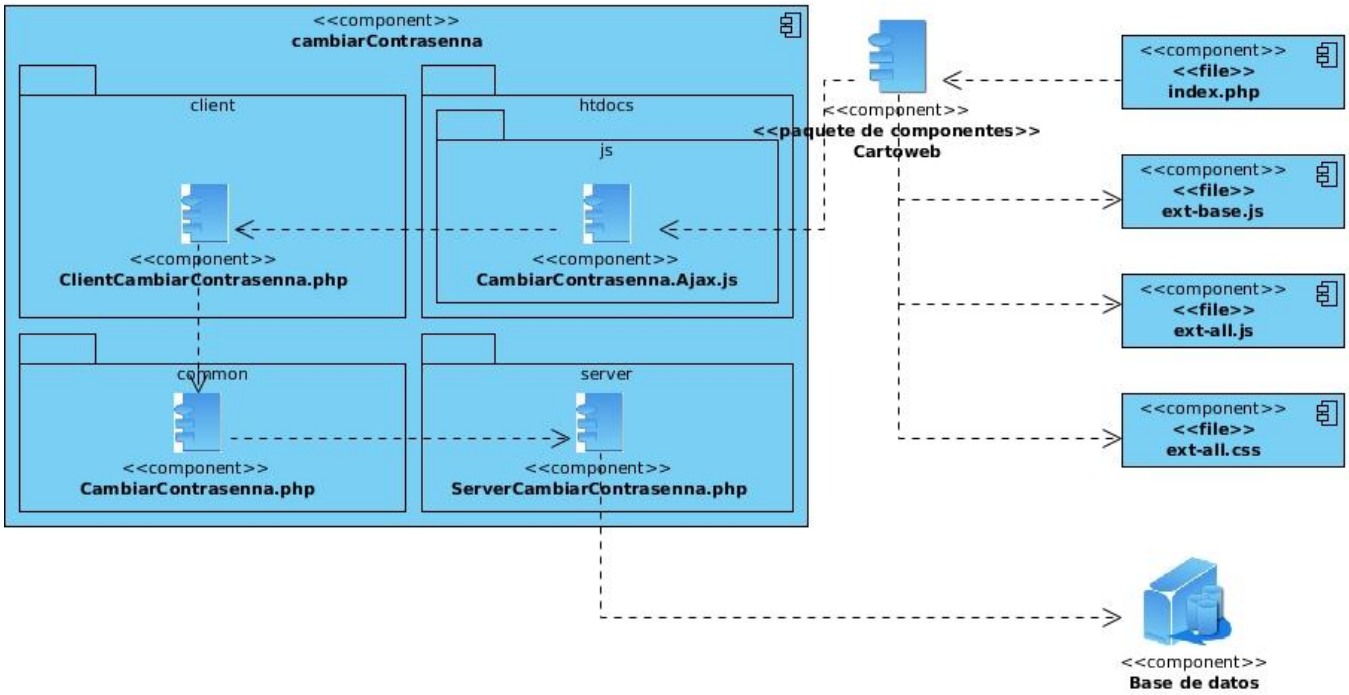

Diagrama de componentes: CU Cambiar Contraseña

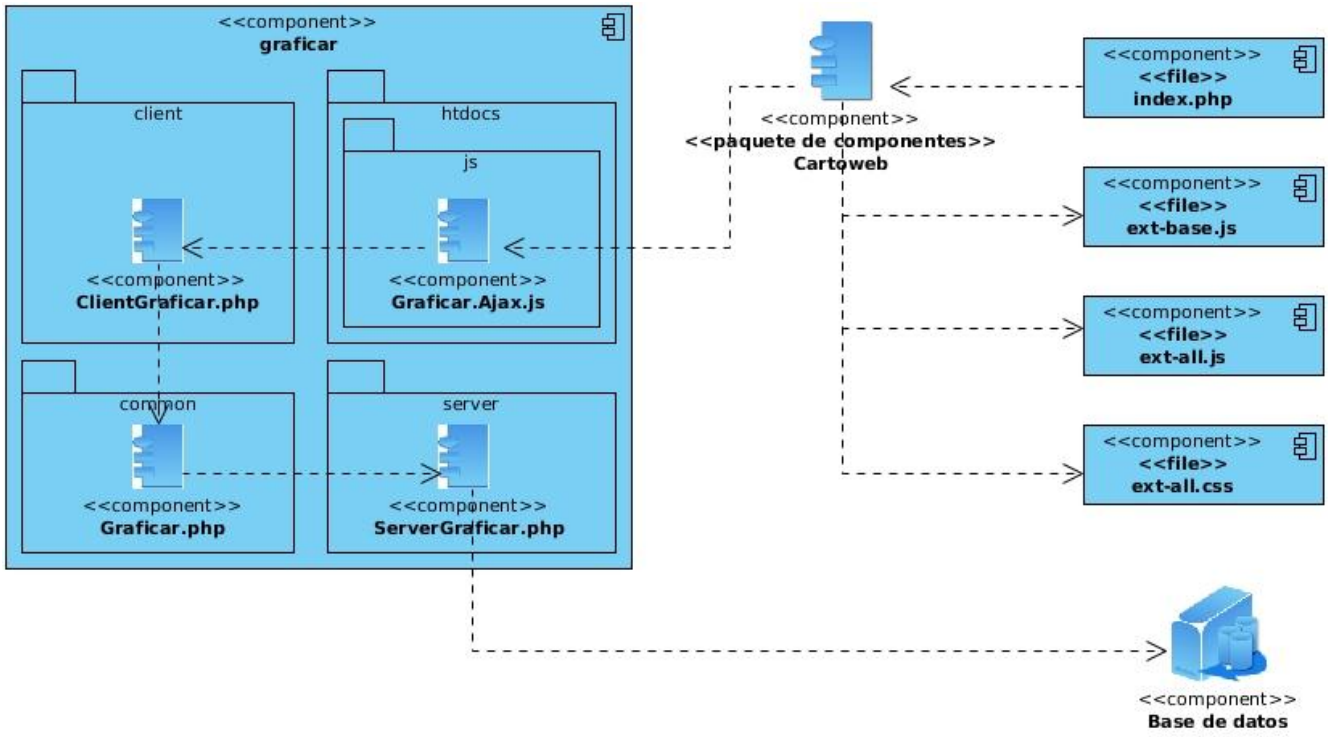

Diagrama de componentes: CU Crear Gráfico

Glosario de Términos

Artefacto: Información que es utilizada o producida mediante un proceso de desarrollo de software.

CASE: (Computer Aided Software Engineering).Constituye una herramienta que ayuda al ingeniero de software a desarrollar y mantener software.

Compilador: Un compilador es un programa que permite traducir el código fuente de un programa en lenguaje de alto nivel, a otro lenguaje de nivel inferior.

Componente de software: Se define típicamente como algo que puede ser utilizado como una Caja Negra, en donde se tiene de manera externa una especificación general, la cual es independiente de la especificación interna.

Datos Espaciales: "Los datos geográficos se definen como cualquier información sobre objetos o fenómenos que tengan una ubicación relativa con respecto a la superficie de la Tierra" (BRIIGGEMANN Y WESTFALEN, 1995).

Multiplataforma: Término usado para referirse a los programas, sistemas operativos, lenguajes de programación, u otra clase de software, que puedan funcionar en diversas plataformas.

Paradigma: Procede del griego *paradeigma,* que significa "ejemplo" o "modelo". En principio, se aplicaba a la gramática (para definir su uso en cierto contexto) y a la retórica (para referirse a una parábola o fábula). A partir de la década del ´60, comenzó a utilizarse para definir a un modelo o patrón en cualquier disciplina científica o contexto epistemológico.

Plugins: Un software plug-in es un complemento para un programa que añade funcionalidad a la misma.

PostGIS: Es un módulo que añade soporte para objetos geográficos a la base de datos objeto-relacional PostgreSQL. PostGIS "habilita espacialmente" el servidor PostgreSQL, permitiendo que sea utilizado como una base de datos de backend espacial para los Sistemas de Información Geográfica (SIG).

Software: "Producto que los ingenieros de software construyen y después mantienen en el largo plazo. Incluyen los programas que se ejecutan dentro de una computadora de cualquier tamaño y

arquitectura, el contenido que se presenta conforme los programas se ejecuten y los documentos, tanto físicos como virtuales, que engloban todas las formas de medios electrónicos" (PRESSMAN, 2005).

Software libre: Es la denominación del software que respeta la libertad de los usuarios sobre su producto adquirido y, por tanto, una vez obtenido puede ser usado, copiado, estudiado, cambiado y redistribuido libremente.