FACULTAD 6

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

Título: “Componente para el análisis del terreno en la Plataforma GeneSIG”

Autor:
César Alemán González
Tutores:
MSc. Grethell Castillo Reyes
Ing.Yoan Mandina Verdecia

Curso 2015-2016

Frase

El éxito es aprender a ir de fracaso en fracaso sin desesperarse.

Winston Churchill

Declaración de autoría

Declaro ser el único autor del trabajo titulado: “Componente para el análisis del terreno en la plataforma GeneSIG” 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 _____.

_____

César Alemán González

Autor

_____ _____

MsC. Grethell Castillo Reyes

Tutor (a)

Ing. Yoan Mandina Verdecia

Tutor

Datos de contacto

Autor

Nombre: César

Apellidos: Alemán González

Teléfono: 55161499

Tutores

Nombre: Grethell

Apellidos: Castillo Reyes

Centro de trabajo: Universidad de las Ciencias Informáticas

Correo electrónico: gcreyes@uci.cu

Año de graduación: 2012

Título de la especialidad: Máster en Informática Aplicada (2015)

Nombre: Yoan

Apellidos: Mandina Verdecia

Centro de trabajo: Universidad de las Ciencias Informáticas

Correo electrónico: ymandina@uci.cu

Año de graduación: 2014

Título de la especialidad: Ingeniería en Ciencias Informáticas

Agradecimientos

Mis más sinceros agradecimientos a La Revolución Cubana y a la Universidad de las Ciencias Informáticas por darme la oportunidad de formarme como joven profesional.

Agradezco a mis padres por apoyarme en todo momento y porque además hoy se hacen realidad sus sueños.

A mi hermano, a mi cuñada y a mi sobrinita por ayudarme a llegar hasta aquí.

A mi esposa Wendy, por estar a mi lado cuando más la necesité.

A mi suegra Dunia, por ofrecerme todo ese cariño incondicional.

A toda mi familia en general por brindarme lo mejor de ellos.

A mis tutores Grethell y Yoan, quienes me supieron guiar de la mejor manera en la realización de este trabajo.

Al tribunal y mi oponente por sus consejos y recomendaciones en aras de lograr un trabajo con la calidad requerida.

A mis amistades de la Universidad, principalmente: Edel, Alejandro, Raúl, Rafa. Carlos y el Cuso por ser mis hermanos incondicionales. En fin a todas aquellas personas que de una forma u otra, hicieron posible la realización de este trabajo: Muchas Gracias.

Dedicatoria

Dedico este Trabajo de Diploma a mis padres, a mi hermano y a mi esposa, por acompañarme en todo momento y brindarme todo el amor y dedicación que he necesitado.

Resumen

Los Sistemas de Información Geográfica (SIG) son sistemas capaces de integrar, almacenar, editar, analizar, compartir y mostrar la información geográficamente referenciada. Estos se están convirtiendo en herramientas indispensables en la toma de decisiones en las que la información espacial tiene una especial relevancia. La plataforma GeneSIG es un SIG y su estructura permite personalizar sus funcionalidades a cualquier negocio que lo requiera a través de la reutilización de sus componentes. La información geográfica con la cual se trabaja en los SIG puede encontrarse en dos tipos de presentaciones o formatos: vectorial y *raster*. El análisis de la información en formato *raster* permite detectar posibles inundaciones, inclinación de una superficie, pendientes existentes, variabilidad de un relieve, entre otros. Actualmente la plataforma GeneSIG no cuenta con un componente para facilitar dichos análisis. Con la presente investigación se desarrolló un componente para la plataforma GeneSIG que facilita el análisis de las características del terreno a partir de la información geográfica almacenada en formato *raster*. Este componente fue construido utilizando la librería GDAL, los lenguajes de programación PHP y JavaScript, el sistema gestor de bases de datos PostgreSQL y la metodología de desarrollo de software Prodesoft. Las pruebas aplicadas al componente arrojaron resultados satisfactorios. La aplicación permite varios tipos de análisis sobre el modelo de presentación *raster*, estos son: mapa de sombra, análisis de pendiente y de orientación, índice de escabrosidad del terreno y de posición topológica.

Palabras claves:

Componente, Sistemas de información geográfica, *raster.*

Abstract

Geographic Information Systems (GIS) are systems capable of integrating, storing, editing, analyzing, sharing, and displaying geographically referenced information. These are becoming indispensable tools in decision-making in which the spatial information is of particular relevance. The GeneSIG is a GIS platform and its structure allows you to customize its functionalities to any business that requires him through the reuse of components. The geographical information with which you work in GIS can be found in two types of presentations and formats: vector and raster. The analysis of information in raster format allows to detect possible floodings, a surface tilt, existing slopes, variability of a relief, among others. Currently the GeneSIG platform does not have a component to facilitate such analyzes. With this research a component for GeneSIG platform that facilitates the analysis of terrain features from the geographical information stored in raster format was developed. This component was built using the GDAL library, PHP and JavaScript programming languages, PostgreSQL database management system and Prodesoft methodology of software development. The test applied to the component was satisfactory. The application allows several types of analysis on the raster model, these are: shadow map, slope analysis and guidance, terrain ruggedness index and topological position

Keywords:

Component, Geographic Information Systems, raster.

Índice de contenido

Índice

Índice de figuras

Índice de tablas

Introducción

El avance de los Sistemas de Información Geográfica (SIG) está muy vinculado a las Tecnologías de la Información y las Comunicaciones (TIC), pues estos sistemas se han ido perfeccionando de manera sofisticada hasta el entorno de nuestros días. También el uso de Internet como fuente de información cartográfica ha sido fundamental para el progreso de estas aplicaciones en la sociedad. Con toda la evolución de la digitalización que rodea el mundo, los SIG han ascendido en concepto de aplicaciones pues han desarrollado gran potencial tecnológico favoreciendo la mejora de algunas destrezas, difíciles de lograr de formas manuales.

El Centro Nacional de Información Geográfica y Análisis define un SIG como “un Sistema de hardware, software y procedimientos elaborados para facilitar la obtención, gestión, manipulación, análisis, modelado, representación y salida de datos espacialmente referenciados para resolver problemas complejos de planificación y gestión”, con el objetivo principal de facilitar la resolución de interrogantes relacionados con información espacial y permitir, a partir de dichas respuestas, la toma de decisiones .

Los SIG permiten hacer un análisis exhaustivo del territorio en los ámbitos más diversos. Son herramientas versátiles, con un amplio campo de aplicación en cualquier actividad que conlleve un componente espacial. Así, la tecnología de los SIG puede ser utilizada para investigaciones científicas, para gestión de los recursos y activos, en arqueología, en evaluación del impacto ambiental, para la planificación urbana, en cartografía, sociología, geografía histórica, marketing o logística, por nombrar sólo algunos ámbitos de aplicación.

Los SIG se están convirtiendo en herramientas indispensables en la toma de decisiones en las que la información espacial tiene una especial relevancia. De alguna de estas decisiones depende en muchos casos el éxito o el fracaso de un negocio o bien la mejora considerable de la productividad de una empresa. Teniendo en cuenta esto resulta fácil comprender la relevancia que estas tecnologías están adquiriendo para el mundo empresarial.

La información geográfica con la cual se trabaja en los SIG puede encontrarse en dos tipos de presentaciones o formatos: la información en formato vectorial donde la información gráfica en este tipo de formatos se representa internamente por medio de segmentos orientados de rectas o vectores. De este modo un mapa queda reducido a una serie de pares ordenados de coordenadas, utilizados para representar puntos, líneas y superficies. La captura de la información en el formato vectorial se hace por medio de: mesas digitalizadoras, convertidores de formato *raster* a formato vectorial, sistemas de

geoposicionamiento global (GPS), entrada de datos alfanumérica, entre otros y la información en formato *raster***,** este formato se obtiene cuando se "digitaliza" un mapa o una fotografía o cuando se obtienen imágenes digitales capturadas por satélites. En ambos casos se obtiene un archivo digital de esa información. La captura de la información en este formato se hace mediante los siguientes medios: scanners, imágenes de satélite, fotografía aérea, cámaras de video entre otros.

Entre los análisis que se realizan sobre el modelo de presentación *raster* se encuentra el mapa de sombra que permite visualizar si el área es plana o accidentada, el análisis de pendientes que se entiende como el desnivel altitudinal entre dos puntos en relación a la distancia que los separa, en su adaptación al cálculo SIG parte de registrar la variación de altitud entre dos puntos, en relación a la distancia que los separa, que se identifica como la distancia entre esos centroides y el análisis de orientaciones que se identifica la dirección que toma cada una de las celdas, consideradas planos inclinados.

En la Universidad de las Ciencias Informáticas (UCI) se encuentra el proyecto “Aplicativos SIG” perteneciente al centro de desarrollo Geoinformática y Señales Digitales (GESYSED) de la Facultad 6, en dicho proyecto se realiza análisis *raster* de la información geográfica a través de la plataforma GeneSIG que permite realizar representaciones y análisis de información referenciada geográficamente. Actualmente dicha plataforma cuenta con los mecanismos para visualizar la información geográfica en formato *raster*. Sin embargo, hasta el momento, los usuarios de GeneSIG deben acudir a otros SIG, para realizar análisis *raster* sobre la información que visualizan, pues la Plataforma no cuenta con las herramientas necesarias para realizar esta tarea. Lo anterior dificulta el proceso de toma de decisiones en GeneSIG cuando se trata de analizar aspectos críticos del terreno como es el caso de posibles inundaciones, la inclinación de una superficie, la determinación de zonas de poca pendiente favorables para la construcción, o zonas de mucha pendiente que determinan erosión o deslizamientos de tierra y en general para determinar la variabilidad de un relieve en un entorno determinado. Estas limitantes desfavorecen la usabilidad del producto, así como el resultado de procesos implicados.

A partir de la situación descrita anteriormente, se presenta como **problema de la investigación:** ¿Cómo facilitar el análisis de las características del terreno en la Plataforma GeneSIG partiendo de la información geográfica almacenada en formato *raster*? Este problema se define con el **objeto de estudio:** Proceso de análisis del terreno sobre información raster el cual está delimitado por el **campo de acción:** Análisis del terreno sobre información raster en Sistemas de Información Geográfica. Para resolver el problema planteado surge como **objetivo general:** Desarrollar un componente para la plataforma GeneSIG que

permita facilitar el análisis de las características del terreno a partir de la información geográfica almacenada en formato *raster*.

Para dar cumplimiento al objetivo planteado, se deben realizar las siguientes **tareas de investigación:**

- Selección de las herramientas, metodología y tecnologías a utilizar en el desarrollo del componente.
- Identificación de los requisitos funcionales y no funcionales para el correcto funcionamiento del componente para el análisis del terreno en la Plataforma GeneSIG.
- Implementación del componente análisis del terreno en la plataforma GeneSIG.
- Realización de pruebas funcionales y otros tipos de pruebas para comprobar el correcto funcionamiento del componente.

Para darle cumplimiento a las tareas anteriores se utilizarán los siguientes **métodos de la investigación científica**:

Teóricos:

Histórico-Lógico: El método histórico estudia la trayectoria real de los fenómenos y acontecimientos en el transcurso de su historia. El método lógico investiga las leyes generales del funcionamiento y desarrollo de los fenómenos (Zayas, 1997). Este método se emplea para analizar la evolución de los conceptos asociados a las herramientas del análisis del terreno.

Analítico-Sintético: Permite la descomposición de un todo complejo en sus partes y cualidades. La síntesis, por su parte, establece la unión entre las partes, previamente analizadas y posibilita descubrir relaciones y características generales entre los elementos de la realidad (Zayas, 1997). Este método se emplea para la valoración de las soluciones existentes que responden al campo de acción. Además se utiliza en la selección de las herramientas y tecnologías que se emplearán en el desarrollo de la solución propuesta.

El presente documento está dividido en tres capítulos a continuación se hace referencia al contenido que se aborda en cada uno de ellos.

Capítulo 1: Fundamentación Teórica

En este capítulo se describen los tipos de análisis del terreno que se efectúan sobre el modelo de presentación *raster* en los SIG y se realiza un análisis de las soluciones existentes. Además se selecciona

Prodesoft como metodología de desarrollo de software a utilizar y se seleccionan las herramientas, técnicas y tecnologías para darle solución al problema.

Capítulo 2: Análisis y Diseño.

En este capítulo se realiza una descripción de la solución propuesta y el modelado del dominio del problema. Además se identifican los requisitos funcionales y no funcionales y se presentan los artefactos obtenidos en el proceso ingenieril.

Capítulo 3: Implementación y pruebas de la solución.

En este capítulo se procede a implementar el sistema confeccionando el diagrama de componente y el diagrama de despliegue. También se abordan y analizan las técnicas de pruebas y se selecciona las pruebas de caja negra utilizando la técnica de partición de equivalencia. Además se aborda un análisis de los resultados obtenido en la aplicación de las pruebas.

Capítulo 1: Fundamentación Teórica

Introducción

En este capítulo se presenta la definición del marco teórico de la investigación. Se exponen los conceptos necesarios para una mejor comprensión del trabajo desarrollado. Incluyendo el análisis de las tecnologías, metodologías, lenguajes de programación y herramientas más idóneas para el desarrollo del tipo de aplicación que se implementará.

Sistemas de Información Geográfica.

Un SIG se puede definir como aquel método o técnica de tratamiento de la información geográfica que permite combinar eficazmente información básica para obtener información derivada. Para ello, contará tanto con las fuentes de información como con un conjunto de herramientas informáticas (*hardware y software*) que facilitará esta tarea; todo ello enmarcado dentro de un proyecto que habrá sido definido por un conjunto de personas, y controlado, así mismo, por los técnicos responsables de su implantación y desarrollo. En resumen, un SIG es una herramienta capaz de combinar información gráfica (mapas) y alfanumérica (estadísticas) para obtener una información derivada sobre el espacio (Bravo, 2000).
Algunas definiciones de Sistema de Información Geográfica son:

- CEBRIÁN (1988): “Una base de datos computarizada que contiene información espacial”.
- GOODCHILD (1985): “Un sistema que utiliza una base de datos espacial para generar respuestas ante preguntas de naturaleza geográfica”.
- ARONOFF (1989): “Un conjunto de procedimientos manuales o computarizados usado para almacenar y tratar datos referenciados geográficamente”.
- BURROUGH (1986): “Un potente conjunto de herramientas para recolectar, almacenar, recuperar a voluntad, transformar y presentar datos espaciales procedentes del mundo real”.
- NCGIA (1990): “Sistema de hardware, software y procedimientos diseñado para realizar la captura, almacenamiento, manipulación, análisis, modelización y presentación de datos referenciados espacialmente para la resolución de problemas complejos de planificación y gestión”.
- STAR y ESTES (1990): “Sistema de Información diseñado para trabajar con datos georreferenciados mediante coordenadas espaciales o geográficas”.

Funcionalidad de los SIG:

Un SIG permite realizar análisis vicariantes, es decir, con el mismo se puede realizar comparaciones entre escalas y perspectivas emulando una cierta capacidad de representación de diferentes lugares al mismo tiempo, diferenciar entre cambios cualitativos y cuantitativos aportándonos una gran capacidad de cálculo y gestionar un gran volumen de información a diferentes escalas y proyecciones; además integra espacialmente datos tabulares y geográficos junto a cálculos sobre variables (topología) y admite multiplicidad de aplicaciones y desarrollos poniendo a disposición herramientas informáticas estandarizadas que pueden ir desde simples cajas de herramientas hasta paquetes llave en mano. Por estos motivos, se puede afirmar que cada vez los SIG son más imprescindible para todas aquellas personas que utilizan información geográfica.

Descripción general de GeneSIG

La Plataforma GeneSIG es un sistema desarrollado para agilizar el desarrollo de SIG en la web. Posee una estructura basada en *plugins*, lo que la convierte en una plataforma con un alto grado de interoperabilidad debido a que permite agregar o quitar componentes de manera sencilla. Es un producto totalmente desarrollado y logrado sobre software libre que permite la personalización de sus funcionalidades a cualquier negocio que lo requiera a través de la reutilización de sus componentes. Brinda servicios de georreferenciación y localización de recursos, así como la inclusión de datos y ubicación de nuevos objetos sobre mapas. Presenta también un abanico bastante completo de características propias de un geoportal, con posibilidad de ir añadiendo o desarrollando nuevos *plugins*. Y es precisamente a través de estos *plugins* agregados de forma convencional, que GeneSIG posee un amplio conjunto de funcionalidades, que actúan como herramientas de la misma plataforma y le brindan la posibilidad de ser altamente modular y escalable (GeneSig, 2012).

Entre sus principales componentes se encuentran:

- Componente de Navegación.
- Componente de Selección.
- Componente de Consulta Espacial.
- Componente de Análisis.

- Componente de Configuración del Mapa.
- Componente de Impresión.
- Componente de Catálogo.
- Componente de Servicios.
- Componente de Edición.
- Componente de Ayuda.

La arquitectura de la plataforma GeneSIG es una arquitectura distribuida. El sistema se basa en la arquitectura cliente–servidor sobre plataforma Web, donde cada instancia del sistema en el cliente es independiente de la ejecución de otra.

Principales funcionalidades de GeneSIG:

- Permitir la representación geoespacial de la información asociada a cualquier negocio que lo requiera.
- Proporcionar servicios de acceso a la información geográfica, para su consulta, análisis y visualización, mediante una interfaz de usuario sencilla y de fácil manejo que pueda ser utilizada por usuarios no especializados en tecnología SIG.
- Integra la información socioeconómica existente (recursos humanos, activos fijos, entidades de servicios, lugares de interés, etc.) con la información geográfica asociada.

Conceptos asociados al dominio de la investigación

A continuación se identifican los conceptos que permiten un mejor dominio del problema, enmarcándose en las características fundamentales de los SIG, para lograr una mejor comprensión del problema y los factores que están vinculados al mismo se describen los siguientes conceptos:

Cartografía

La Cartografía es la disciplina que integra la ciencia que se encarga del estudio y de la elaboración de los mapas (Real Academia Española, 2016).

Formato *raster*

El formato *raster* se fundamenta en la división del área de estudio en una matriz de celdillas, generalmente cuadradas. Cada una de estas celdillas recibe un único valor que se considera representativo para toda la superficie abarcada por la misma. Este formato, por tanto, cubre la totalidad

del espacio, este hecho supone una ventaja fundamental ya que pueden obtenerse valores de forma inmediata para cualquier punto del mismo (GIS Modeling in Raster, 2002).

Elementos que componen una capa *raster*

Una capa en formato *raster* está compuesta por cuatro elementos fundamentales:

- La matriz de datos, que puede contener tres tipos de datos:
  - Valores numéricos en caso de que la variable representada sea cuantitativa.
  - Identificadores numéricos en caso de que se trate de una variable cualitativa. Estos identificadores se corresponden con etiquetas de texto que describen los diferentes valores de la variable cualitativa.
  - Identificadores numéricos únicos para cada una de las entidades representadas en caso de que la capa *raster* contenga entidades (puntos, líneas o polígonos).
- Información geométrica acerca de la matriz y de su posición en el espacio:
  - Número de columnas (nc).
  - Número de filas (nf).
  - Coordenadas de las esquinas de la capa (e, w, s, n).
  - Resolución o tamaño de pixel en latitud (rx) y en longitud (ry).
- Una tabla de colores que permita decidir de qué color se pintará cada celdilla en la pantalla.
- En caso de que la variable sea cualitativa, una tabla que haga corresponder a cada identificador numérico una etiqueta de texto descriptiva.

Hay una serie de convenciones acerca de la forma de representación. Así la primera columna en una capa *raster*, la columna número cero, es la de la izquierda (oeste) aumentando el número de columna hacia la derecha (este), mientras que la primera fila, la número cero, será la superior (norte) aumentando hacia abajo (sur), en sentido contrario al que siguen las coordenadas geográficas y las representaciones en un modelo vectorial. Puesto que la primera fila es la número 0, estas se numeran desde 0 hasta nf – 1, y lo mismo para las columnas (GIS Modeling in Raster, 2002).

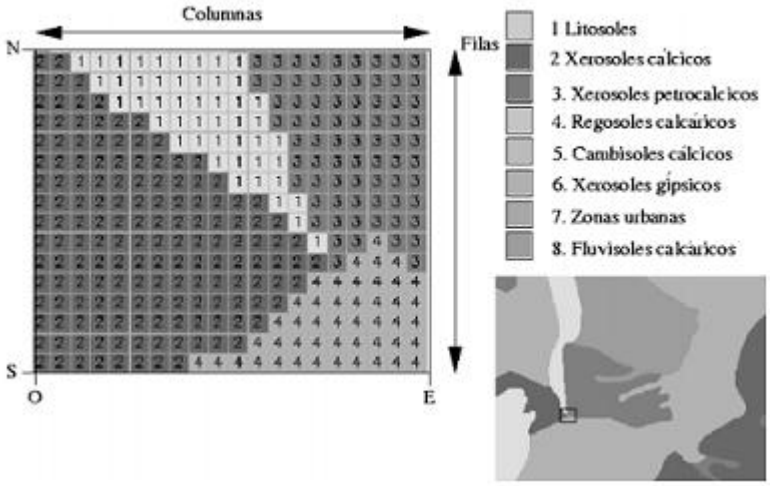

FIG. 1: CODIFICACIÓN DE UNA VARIABLE CUALITATIVA EN FORMATO RASTER.

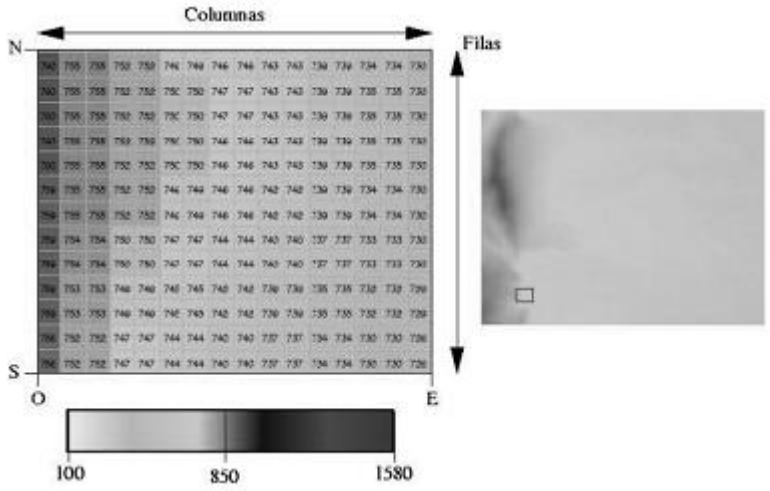

FIG. 2: CODIFICACIÓN DE UNA VARIABLE CUANTITATIVA EN FORMATO RASTER.

Modelos Digitales del Terreno (MDT)

Se han definido como un conjunto de datos numéricos que describe la distribución espacial de una característica del territorio (Felicísimo,A.M, 1994).

Aplicaciones de los MDT:

- **Apoyo en análisis estadísticos**

Las variables incluidas en un MDT son factores de gran importancia en un gran número de procesos ambientales (precipitación, insolación-temperatura, flujos hídricos, erosión, distribución de hábitats, etc.) por tanto van a ser un elemento clave a la hora de estimar otras variables mediante procedimientos de interpolación global por regresión.

El procedimiento básico sería medir en diferentes puntos del área de estudio la variable independiente junto con las variables dependientes incluidas en el MDT. Un análisis posterior generaría un modelo de regresión de tipo Y = F (Z, S, O, Ct,..). Puesto que se dispone de capas *raster* para cada una de estas variables, resultará fácil llevar a cabo la interpolación por regresión.

Modelos climáticos

La topografía es el principal factor local que limita la energía solar incidente sobre la superficie terrestre. La variedad de altitudes, pendientes y orientaciones crean fuertes contrastes locales que afectan directa e indirectamente a procesos biológicos y físicos. Algunos de estos factores son modelizables con los MDT.

La existencia de zonas de sombra es una variable de gran interés en regiones montañosas, donde el relieve puede ser el factor determinante más importante del clima local. Se define la insolación potencial en un punto como el tiempo máximo que ese lugar puede estar sometido a la radiación solar directa en ausencia de nubosidad. La insolación potencial depende directamente del ángulo de incidencia del sol respecto a la superficie terrestre y del ocultamiento topográfico ante una trayectoria concreta del Sol. La insolación potencial se expresa en unidades energía partido por espacio y se refiere a un instante concreto. Puede, sin embargo, integrarse para períodos de tiempo mayores. La relación entre cada celda y la superficie de referencia se realiza mediante un índice de exposición definido como el cociente entre la radiación solar incidente sobre un lugar del terreno y la superficie de referencia. El cálculo de los índices de exposición da un parámetro cuantitativo útil a la hora de comparar las condiciones ambientales en una zona determinada ya que representan una medida objetiva de las diferencias dentro del área del modelo basada exclusivamente en criterios geométricos. El ángulo solar resulta además de utilidad en aplicaciones relacionadas con la teledetección (corrección por iluminación y cálculo de reflectividades).En cuanto a la radiación recibida se utiliza en:

1. Modelos de estimación de variables climáticas (temperatura, evapotranspiración).
2. Modelos de distribución potencial de especies animales o vegetales.

Modelos hidrológicos

La superficie terrestre constituye la base sobre la que tienen lugar, y que por tanto condiciona, gran parte de los procesos de transferencia de materia y energía que tienen lugar sobre la superficie terrestre. La

disponibilidad de un modelo de dicha superficie permite simular estos procesos, con lo que se consigue experimentar independientemente del sistema real. La simulación permite obviar los riesgos inherentes a la experimentación, alcanzar una completa independencia temporal, repetir el experimento un número de veces arbitrario. Las características topográficas de una ladera determinan las pautas por las cuales el agua circula sobre ella. El modelo digital de elevaciones contiene información suficiente para definir, al menos en una primera aproximación, las propiedades de la red de drenaje superficial y, por extensión, de la cuenca hidrológica. Se denomina línea de flujo al trayecto que, a partir de un punto inicial, seguiría la escorrentía superficial sobre el terreno Las líneas de flujo siguen la línea de máxima pendiente por lo que pueden deducirse del modelo digital de pendientes con las únicas limitaciones que las derivadas de la calidad del MDE original. A partir del trazado de las líneas de flujo es posible definir la red hidrológica, el área subsidiaria de una celda y, por extensión, las cuencas hidrológicas: Se define el área subsidiaria de una celda como el conjunto de celdas cuyas líneas de flujo convergen en ella; una cuenca hidrológica está formada por el área subsidiaria de una celda singular, que actúa como sumidero. La magnitud del área subsidiaria de una celda del MDE está directamente relacionada con el caudal máximo potencial, CMP, en el mismo. En efecto, el caudal que puede circular en un momento dado en un punto del terreno depende, entre otros factores, de la magnitud del área subsidiaria, de las precipitaciones sobre ella y de la pendiente de la zona, que permite la circulación con menor o mayor rapidez. En función de estos parámetros es posible simular el CMP en un modelo digital del terreno. Otra información de gran interés hidrológico directamente extraíble de un MDT son las redes de drenaje. Para ello se parte de la hipótesis de que hay un valor umbral de área subsidiaria por encima del cual el cauce en cuestión puede considerarse como perteneciente a un cauce. Por tanto basta con reclasificar el mapa de áreas subsidiarias para asignar un valor 1 a aquellas celdillas con área subsidiaria mayor que dicho umbral y valor 0 o nulo a las restantes. Finalmente, si se quiere el mapa de redes de drenaje en formato vectorial se deberá realizar el correspondiente cambio de formato. Las redes de drenaje extraídas con este procedimiento presentan algunas deficiencias:

- El que en una determinada celdilla se inicie un cauce depende no sólo de su área subsidiaria sino también de las características litológicas e incluso de uso del suelo de la misma. Por tanto utilizar un sólo valor umbral para todo el área de trabajo resulta bastante simplista.
- Debido al algoritmo que genera las direcciones de flujo y los mapas de área subsidiaria, los cauces resultantes tienden a adoptar un carácter rectilíneo.

La modelización hidrológica basada en modelos digitales de terreno pretende estimar los caudales generados en una cuenca a partir de sus características topográficas así como las áreas inundables en función de la altura esperable de las láminas de agua. Evidentemente, es necesario compaginar los resultados obtenidos a partir de los modelos de elevaciones con estimaciones de la capacidad de infiltración de los suelos o la estimación de precipitaciones máximas esperables.

Modelos de visibilidad

Los modelos de visibilidad establecen el área que se puede ver desde un punto y, por tanto, el área desde la que puede verse ese punto. El primer caso puede ser útil para el diseño de redes de control (de incendios forestales por ejemplo), el segundo como criterio a la hora de ubicar infraestructuras desagradables (vertederos). Dos puntos serán mutuamente visibles si la línea recta que los une tiene siempre una altitud superior a la del terreno. La cuenca visual de un punto base sería entonces el conjunto de puntos de un MDE que son mutuamente visibles con dicho punto base. El análisis de cuencas visuales puede utilizarse para la evaluación del impacto visual de actuaciones con efectos negativos sobre el paisaje. Es posible construir un modelo de visibilidad, donde cada punto tiene asignado un valor proporcional a la extensión de su cuenca visual. Un modelo de este tipo puede servir de base objetiva para la toma de decisiones ya que permite conocer y comparar con fiabilidad la incidencia visual de las alternativas existentes.

Modelos Digitales de Elevación (MDE)

Se define como una estructura numérica de datos que representa la distribución espacial de la altitud de la superficie del terreno.

Fórmula de Zevenbargen&Thorne

Los umbrales topográficos que caracterizan la aparición del flujo concentrado y la generación de cárcavas en las laderas para un mismo tipo, uso y manejo del suelo han sido estudiados por Thorne y Zevenbergen. Todos los índices o modelos topográficos usaron una relación similar a la ecuación 1: $A_d^b S Pl^k > I$ (1) donde, A_d es el área de desagüe aguas arriba del punto considerado; *b* es el exponente relativo al área; *S* es la pendiente local aguas arriba; *Pl* es la curvatura plana local, no considerada en algunas expresiones (exponente *c*=0); *I* es el umbral crítico a partir del cual se produce un flujo concentrado cuyo esfuerzo cortante es superior a la tensión crítica de la superficie del suelo, que genera la incisión y el desarrollo de las cárcavas (Taguas, 2010).

Tipos de Análisis sobre el modelo de datos *raster*

Existen varios tipos de análisis que se realizan sobre el modelo de datos *raster*, entre ellos los más utilizados son los de pendiente y orientación.

Análisis de pendientes

Las pendientes derivadas mediante análisis SIG *raster* consisten en una adaptación del cálculo conocido de la pendiente por métodos convencionales. La pendiente, entendida como el desnivel altitudinal entre dos puntos en relación a la distancia que los separa, en su adaptación al cálculo SIG parte de registrar la variación de altitud entre dos puntos -que en este caso son los centroides del pixel-, en relación a la distancia que los separa, que se identifica como la distancia entre esos centroides. El análisis de la pendiente en un SIG es una operación de vecindad inmediata y capa única.

En el cálculo de la pendiente de cada celda de un Modelo Digital de Elevación (MDE) de partida intervienen los desniveles altitudinales entre la celda de referencia y sus vecinas, por el lado o bien por el lado y el vértice. De los cuatro o, en su caso, ocho cálculos de pendientes entre cada celda y sus vecinas el mapa de pendientes resultante asume en cada celda un único valor que puede ser asignado en función de diferentes métodos, entre los que destaca el de la "máxima pendiente" según el cual en el mapa resultante se muestra únicamente el máximo valor de pendiente obtenido tras los cálculos detallados de vecindad inmediata. El resultado podría expresarse en diferentes unidades, siendo las más utilizadas grados (ángulo formado entre el plano inclinado y horizontal) y porcentaje (de Cos Guerra, 2012).

Análisis de orientaciones

Como orientación se identifica la dirección que toma cada una de las celdas, consideradas planos inclinados. Inicialmente, la orientación a partir de la rosa de los vientos se puede expresar en términos cualitativos o bien cuantitativos. En el caso de los mapas de orientaciones generados con un SIG los valores de orientaciones se expresan de forma numérica, en grados azimut (360º a partir del Norte que es 0º). En grados, la correspondencia del rango numérico con la interpretación cualitativa es la siguiente: 45º - Noreste, 90º - Este, 135º - Sureste, 180º - Sur, 225º - Suroeste, 270º - Oeste, 315º - Noroeste y, finalmente, 360º ó 0º - Norte.

Un aspecto importante a considerar en los modelos de orientaciones derivados es el que hace referencia a los pixeles llanos (de pendiente 0, planos no inclinados). Normalmente son tratados de forma especial por los SIG en el cálculo del mapa de orientaciones y con frecuencia no toman ninguna orientación concreta -y todas a la vez-, asumiendo en el modelo derivado un valor que se sale del rango lógico de los

grados de orientación señalados anteriormente. Este valor correspondiente a todas las orientaciones es anómalo en relación a los grados de orientación (rango 0-360º) y pueden aparecer con el valor -1, o bien 99999, etc. dependiendo del programa SIG utilizado.

Dado que el análisis de la orientación se puede estimar tomando como referencia únicamente el MDT, se trata de una operación espacial de capa única y, dado que intervienen las celdas contiguas a la de referencia, se considera una operación espacial de vecindad inmediata (de Cos Guerra, 2012).

Soluciones existentes

El acceso al conocimiento geográfico más fácil y disponible, ha encauzado a muchos especialistas del mundo, a buscar alternativas para mejorar las tecnologías SIG y con ello el tratamiento con los mapas, que son la fuente de la información geográfica. La forma de crear, mostrar, acceder, compartir e interactuar con un mapa SIG, está en dependencia de las características específicas de un sistema. Es de interés conocer durante la investigación, la capacidad que tienen los SIG de realizar estas operaciones, para ello se hace necesario realizar un análisis de las propiedades de estos y las ventajas que proporciona su uso.

Para la realización del análisis, se tienen en cuenta las diferentes formas de representación de las tecnologías SIG, específicamente los SIG de escritorio. Existen varios clientes SIG de escritorio entre los que se encuentra los tres software SIG con más relevancia del momento: ArcGIS, gvSIG y Quantum GIS. Los tres tienen sus ventajas y sus inconvenientes.

GvSIG Desktop:

Es un potente Sistema de Información Geográfica libre, diseñado para dar solución a todas las necesidades relacionadas con el manejo de información geográfica. Se caracteriza por ser una solución completa. Es capaz de acceder a los formatos más comunes, como vectoriales, *rasters*, locales y remotos, integra estándares de Open Geospatial Consortium (OGC), y cuenta con un amplio número de herramientas para trabajar con información de naturaleza geográfica (consulta, creación de mapas, geoprocesamiento, redes, etc.) que lo convierten en una herramienta ideal para usuarios que trabajen con la componente territorial. Funciona en distintas plataformas Linux, Windows y Mac OS. Se distribuye bajo la licencia GNU/ GPL, lo que permite su libre uso, distribución, estudio y mejora (Asociacion gvSIG, 2009).

ArcGIS Desktop:

Es un complejo Sistema de Información Geográfica para compilar, usar y administrar la información geográfica. Incluye un conjunto de aplicaciones: ArcMap, ArcCatalog, ArcGlobe, ArcScene, ArcToolbox y ModelBuilder, que admiten diversas tareas SIG, incluidas la representación cartográfica, la compilación de datos, el análisis, la administración de geodatabases y el uso compartido de información geográfica. Es compatible con diversos sistemas operativos como Windows, GNU/Linux y Unix. Las actividades claves que realizan los usuarios en ArcGIS Desktop son el trabajo con los mapas, análisis espacial y compilación de datos. Los mapas son esenciales, porque hacen que toda la información cobre vida y son el mecanismo utilizado para editar y proporcionar análisis espacial a muchos usuarios (ENRI, 1995).

QGIS:

Es una aplicación SIG de escritorio, con grandes potencialidades para la edición de mapas, de código abierto. El proyecto nació en mayo del 2002 y se estableció como un proyecto dentro del *Source forge* (es una central de desarrollos de software que controla y gestiona varios proyectos de *software* libre y actúa como un repositorio de código fuente) en junio del mismo año. Actualmente QGIS corre en la mayoría de las plataformas como Unix, Windows y OSx. Esta desarrollado utilizando el Qt y como lenguaje C++. Esto hace que QGIS sea rápido y tenga una interfaz amigable para los usuarios. QGIS se ha diseñado con una arquitectura de complementos. Como complemento de análisis del terreno que realiza QGIS se encuentra (MappingGIS, 2015):

Mapa de sombras: Crea un mapa de sombra utilizando la luz y sombra que proporciona un aspecto más tridimensional para un mapa de relieve sombreado.

Pendiente: Calcula el ángulo de la pendiente de cada celda en grados (basado en primer orden estimación derivada).

Orientación: Exposición (iniciar con 0 para la dirección norte, en grados anti horario).

La tecnología gvSIG desktop presenta las características idóneas para resolver el problema planteado en la investigación, como por ejemplo que esta herramienta se distribuye bajo la licencia GPL, permitiendo su uso, la libre distribución y modificación, cumpliendo con la política de migración al *software* libre que impulsa el país, es compatible con los principales sistemas operativos como Windows y Linux. Sin embargo, la forma en que realizan el proceso de gestión de las vistas de los mapas, no responde a la necesidad que tiene la plataforma GeneSIG, porque ya en ésta, los mapas están creados, solamente se necesita guardar una ubicación geográfica que ha sido seleccionada por el usuario. En el caso de ArcGIS desktop, es una potente herramienta que sí da respuesta a la necesidad de la plataforma. Incluso permite compartir carpetas con las vistas de los mapas entre varios usuarios, pero es muy costosa, es un software

propietario y las licencias limitan una mejor distribución del sistema SIG en los ordenadores. De la herramienta QGIS surgieron la idea de la interfaz principal y la mayoría de los requisitos funcionales y no funcionales de la aplicación ya que esta herramienta era la que se utiliza en el proyecto y sus funcionalidades son similares a las de la herramienta que se desea construir.

Herramientas y Tecnologías

Para el desarrollo del componente se realiza un análisis de las metodologías, tecnologías, lenguajes de programación y herramientas más idóneas para el desarrollo del componente que se implementará.

1.1 Metodología de desarrollo

Para facilitar el desarrollo de software en un grupo de trabajo y garantizar la calidad se utilizan las metodologías de desarrollo. Una metodología de desarrollo de software es un marco de trabajo que muestra los pasos que se requieren para construir un software de alta calidad. Indicando además qué personas deben participar en el desarrollo de las actividades y qué papel deben cumplir. Guía a los desarrolladores en la realización de las actividades, durante todo el proceso de creación de un producto. Se clasifican en dos tipos: robustas que son usadas en grandes proyectos, proponen un ciclo de desarrollo con mayor énfasis en la planificación y control del proyecto. Las ágiles se utilizan para proyectos de corto plazo con menor tamaño, se caracterizan por tener la capacidad de proveer respuestas rápidas y ser adaptables al cambio. Proporcionan una serie de pautas y principios junto a técnicas pragmáticas, que puede que no curen todos los males, pero harán la entrega del proyecto menos complicada y más satisfactoria tanto para los clientes como para los equipos de entrega (Delavaut, 2012).

1.1.1 ProDeSoft (Proceso de Desarrollo de Software)

En correspondencia con las líneas bases del proyecto “GeneSIG” se define ProDeSoft como proceso de desarrollo de software a emplear en la solución que se propone. El cual se basa en la combinación de varios modelos de procesos: el basado en componentes, el iterativo y el incremental. Este proceso se encuentra disponible en su versión 1.5 y según su especificación (UCID, 2012), plantea que el ciclo de vida de un proyecto está compuesto por cinco fases: inicio, modelación, construcción, explotación experimental y despliegue (ver Figura 3).

FIG. 3: CICLO DE VIDA DE UN PROYECTO, PRODESOFT.

Durante la fase de Inicio se logra una visión preliminar de la problemática a resolver y se definen los recursos relevantes para la ejecución del proyecto. Es decir, se describen los objetivos y el alcance del proyecto, se identifican los involucrados y ejecutores (entidades involucradas), se estima de manera general las actividades a realizar durante todo el ciclo de desarrollo del proyecto (Cronograma General), se establece la estrategia a seguir para realizar la modelación del negocio y la captura de requisitos y de ser necesario se estiman los recursos materiales que deberán ser adquiridos.

En la fase de Modelación se capturan las partes esenciales del sistema, donde se identifican los procesos de negocio fundamentales y se aceptan los requisitos funcionales, obteniéndose la línea base de la arquitectura y una estrategia de construcción de la aplicación aprobada por los implicados en el proyecto. El hito fundamental de esta fase es la liberación de la arquitectura de sistema, datos y despliegue.

En la fase de Construcción se aclaran los requisitos restantes y se completa el desarrollo del sistema sobre una base estable de la arquitectura. Las fases anteriores sólo dieron una arquitectura básica que es aquí refinada de manera incremental conforme se construye el producto. En esta fase todas las características, componentes, y requisitos deben ser integrados, implementados, y probados en su totalidad, obteniendo una versión liberada del producto.

Durante la fase de Explotación Experimental se convierte la versión liberada del producto en una solución estable, donde se eliminan los errores que surgen durante las pruebas y se obtiene una certificación funcional y de seguridad del producto.

En la fase de Despliegue se instala y configura el sistema para un ambiente de producción real, se capacita al personal que usará la aplicación y se continúa dando soporte durante la explotación del sistema, culminando de ser preciso con transferencias tecnológicas (Méndez, 2011).

1.2 Lenguaje de Modelado. UML.

El Lenguaje de Modelación Unificado (UML en inglés *Unified Model Lenguaje*) es un lenguaje para visualizar, especificar, construir y documentar los artefactos de un sistema software. El modelado visual ayuda a mejorar la capacidad del equipo para gestionar la complejidad del software. Combina lo mejor de los métodos de análisis y diseño anteriores a él. Es un lenguaje de modelado de sistemas orientado a objetos de notación para el desarrollo de software. Se estandarizó su uso en la industria de desarrollo de software (Medoly Y. Ivory-Ndiaye, 2016).

1.2.1 Herramienta CASE. Visual Paradigm.

Visual Paradigm para UML (VP - UML) es una herramienta Ingeniería de Software Asistida por Computadora (CASE en inglés Computer Aided Software Engineering). Ofrece un paquete completo necesario para la captura de requisitos, la planificación del software, planificación de pruebas, modelado de clases y modelado de datos. Esta herramienta permite aumentar la calidad del software, a través de la mejora de la productividad en el desarrollo y mantenimiento del software. También permite la reutilización del código, portabilidad y estandarización de la documentación. Además del uso de las distintas metodologías propias de la Ingeniería del Software (Medoly Y. Ivory-Ndiaye, 2016).

Algunas características de Visual Paradigm:

- Generación de bases de datos. Transformación de diagramas de Entidad-Relación en tablas de base de datos.
- Proporciona a los desarrolladores una plataforma que les permite diseñar un producto rápidamente y con la calidad requerida.
- Facilita la interoperabilidad con otras herramientas CASE y se integra con múltiples herramientas de desarrollo, como Eclipse/IBMWebSphere, Jbuilder, NetBeans IDE, Oracle Jdeveloper.

- Genera código y realiza ingeniería inversa para diez lenguajes de programación, entre ellos Java, C++, PHP y XML Schema.
- Genera documentación para el proyecto en HTML, MS Word y PDF.
- Licencia: gratuita y Comercial (Medoly Y. Ivory-Ndiaye, 2016).

1.3 Lenguaje de programación

Los lenguajes de programación son la vía de comunicación entre el hombre y la computadora. Un lenguaje de programación es una técnica estándar de comunicación. Permite expresar las instrucciones que han de ser ejecutadas en una computadora. Consiste en un conjunto de reglas sintácticas y semánticas que definen un programa informático (Bardales, 2007).

1.3.1 Lenguaje de programación del lado del servidor. PHP.

PHP es un lenguaje de programación del lado del servidor gratuito e independiente de plataforma, rápido, con una gran biblioteca de funciones y mucha documentación. Un lenguaje del lado del servidor es aquel que se ejecuta en el servidor web, justo antes de que se envíe la página a través de Internet al cliente. Las páginas que se ejecutan en el servidor pueden realizar accesos a bases de datos, conexiones en red y otras tareas para crear la página final que verá el cliente. El cliente solamente recibe una página con el código HTML resultante de la ejecución de PHP. Como la página resultante contiene únicamente código HTML, es compatible con todos los navegadores. Una de sus grandes cualidades es su versatilidad al momento de escribir código, su sencillez en la sintaxis e inclusive su seguridad; es gratuito y fácil de aprender. Posee una gran variedad de funciones que pueden ser utilizadas para mejorar el rendimiento de los programas y es un lenguaje de uso muy común en la web. Es completamente expandible. (Dave W. Mercer, 2015).

1.3.2 Lenguaje de Programación del lado del cliente. Java Script.

Un lenguaje del lado cliente es totalmente independiente del servidor, lo cual permite que la página pueda ser albergada en cualquier sitio. El código, tanto del hipertexto como de los scripts, es accesible a cualquiera y ello puede afectar a la seguridad.

Java Script es un lenguaje de programación que se utiliza principalmente para crear páginas web dinámicas que permitan intercambiar con los usuarios. Se implementa como parte de un navegador web

permitiendo mejoras en la interfaz de usuario y páginas web dinámicas. Es interpretado, por lo que no es necesario compilar los programas para ejecutarlos. En otras palabras, los programas escritos con Java Script se pueden probar directamente en cualquier navegador que lo soporte, sin necesidad de procesos intermedios (INSTITUTO TECNOLÓGICO DE VERACRUZ, 2011).

El lenguaje de programación Java Script permite que:

- Los programas escritos en este lenguaje no requieren de mucha memoria ni tiempo adicional de transmisión, por ser pequeños y compactos.
- No requiere un tiempo de compilación, pues los scripts se pueden desarrollar en un período de tiempo relativamente corto.
- Es independiente del hardware o sistema operativo, este funciona correctamente siempre y cuando exista un navegador que lo soporte.
- Asegura la permanencia de una operación realizada y aunque falle el sistema esta no podrá deshacerse.

1.5 Entorno de desarrollo integrado (IDE). NetBeans.

NetBeans es un entorno de desarrollo visual de código abierto. Está disponible para diversos sistemas operativos como Solaris, Windows, MacOS y GNU Linux. Permite a los desarrolladores crear rápidamente aplicaciones web, de escritorio y móviles utilizando la plataforma Java. Soporta lenguajes como PHP, JavaScript y AJAX, Groovy y Grails, y C / C + +. Es un producto gratuito y no tiene restricciones de uso, tiene una comunidad que le ofrece soporte a nivel mundial. Permite que las aplicaciones sean desarrolladas a partir de componentes, lo cual posibilita que se puedan construir aplicaciones extensibles. Netbeans incluye herramientas de esquemas XML, orientación a servicios Web y modelado UML (Oracle Corporation, 2013).

1.6 Marco de trabajo de desarrollo.

Un Marco de trabajo [Framework] es una abstracción en el que se brinda al usuario funcionalidades genéricas, proporcionando así software reutilizable utilizado para desarrollar aplicaciones, productos y soluciones. Incluye programas de apoyo, compiladores, bibliotecas de código, una API y conjuntos de herramientas que reúnen a todos los diferentes componentes para permitir el desarrollo de un proyecto o solución. Aumenta la facilidad del trabajo. Promueve buenas prácticas de desarrollo como el uso de patrones (Frederick, 2008).

1.6.1 Marco de trabajo de desarrollo del lado del cliente. ExtJS

ExtJS es una librería Java Script que permite crear aplicaciones web interactivas del lado del cliente, para ello cuenta con una serie de componentes. Es ligera y de alto rendimiento, compatible con la mayoría de navegadores, evitando tener que validar el código para que funcione correctamente en cada navegador. Permite crear aplicaciones complejas utilizando componentes predefinidos. Empezó siendo un conjunto de librerías y extensiones para Interfaz de usuario Yahoo! [Yahoo! User Inteface], (YUI), librería desarrollada en JavaScript que explota las potencialidades del JavaScript Asincrónico y Xml [Asynchronous JavaScript And XML], (AJAX), para el desarrollo de Aplicaciones Ricas de Internet [Rich Internet Applications], (RIA). Con el tiempo se convirtió en un Marco de trabajo independiente y a principios de 2007 se creó una compañía para comercializar y dar soporte del Marco de trabajo ExtJS (Frederick, 2008).

Principales características de ExtJS:

- Alto rendimiento en ejecución debido a la optimización de código Java Script.
- Controles de usuario personalizables.
- Modelo orientado a componentes, bien diseñado y extensible.
- Posee una API intuitiva y fácil de utilizar.
- Es distribuido bajo licencias libres y comerciales.

1.7 Gestor de base de datos. PostgreSQL.

PostgreSQL es un sistema gestor de base de datos objeto-relacional bajo licencia libre y gratuita. En sus últimas versiones posee muchas características que solo se podían ver en productos comerciales. Utiliza un modelo cliente/servidor y además usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando.

Soporte nativo para los lenguajes más populares del medio: PHP, C, C++, Perl, Python. Incluye todas las características de una base de datos profesional (triggers, store procedures, funciones, secuencias, relaciones, reglas, tipos de datos definidos por usuarios, vistas, vistas materializadas) (PostgreeSQL, 2013).

Principales ventajas:

- Permite la gestión de diferentes usuarios, como también los permisos asignados a cada uno de ellos.

- Posee una gran escalabilidad, haciéndolo idóneo para su uso en sitios Web que atienden un gran número de solicitudes.
- Puede ser instalado un número ilimitado de veces sin temor de sobrepasar la licencia.
- Posee estabilidad y confiabilidad reconocida.
- Es extensible a través del código fuente disponible sin costos adicionales.
- Es multiplataforma, disponible en la mayoría de los sistemas operativos.
- Permite implementar reglas, vistas, disparadores, sub-consultas y funciones.

1.7.1 PostGIS: la extensión geográfica de PostgreSQL

PostGIS está creado por Refractions Research Inc, como un proyecto de investigación de tecnologías de bases de datos espaciales. Está publicado bajo licencia GNU. PostGIS es una extensión al SGBD PostgreSQL que permite el uso de objetos SIG. PostGIS incluye soporte para índices GiST basados en R-Tree, y funciones básicas para el análisis de objetos SIG. Esta extensión soporta objetos SIG de características simples, tales como puntos, líneas, polígonos, multilíneas, multipuntos, y colecciones geométricas (PostGIS 2.0 Manual).

1.8 Geospatial Data Abstraction Library (GDAL)

GDAL es una biblioteca de acceso a datos para la lectura y escritura de una gran variedad de formatos de datos geoespaciales. Presenta un modelo abstracto de datos único para todos los formatos soportados. Viene con una gran variedad de utilidades de líneas de comandos para la traducción y procesamiento de datos, que son especialmente útiles para los usuarios finales, en contraste con los programadores que utilizan la biblioteca (Warmerdam, 2008). La biblioteca se divide en un medio *raster* y uno vectorial, cada uno con su propio modelo de datos y API[1]. Se selecciona GDAL v1.10 pues es ampliamente utilizado en el mundo geoespacial de código abierto incluyendo, pero no limitado, a paquetes como MapServer y QGIS. También se utiliza en diversos grados por varios productos de software propietario, incluyendo ArcGIS (Warmerdam, 2008).

[1] *Application Programming Interface* (API, por sus siglas en inglés, Interfaz de Programación de Aplicaciones) es el conjunto de subrutinas, funciones y procedimientos que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

1.9 Conclusiones

En este primer capítulo se definieron una serie de conceptos asociados al problema científico, lo que posibilita en el futuro, un mejor entendimiento y una buena comprensión de la presente investigación. La situación problemática fue expuesta de la forma más concisa y centrada posible, en aras de lograr resultados positivos que mejoren la situación actual en la plataforma GeneSIG. El estudio de las aplicaciones existentes a nivel internacional, aunque ninguna representa una solución al problema a resolver, por la característica de ser sistemas *desktop* y lo que se desea desarrollar debe ser aplicado a un sistema web, sirvió de base para elaborar una idea general de la forma en que se realiza el análisis del terreno, en los sistemas SIG y las ventajas que proporciona el desarrollo de un componente que permita ejecutar dicho proceso.

Introducción

En el capítulo se describen los principales conceptos de entorno que serán objeto de análisis para la realización de la fase de Análisis y Diseño del componente, a partir del modelo de dominio. Se identifican los requisitos funcionales y no funcionales a tener en cuenta para la implementación del componente.

2.1. Modelo de Dominio.

El modelo dominio es una representación visual de los conceptos u objetos del mundo real significativos para un problema o área de interés. Se representa en UML como un diagrama de clases en el que se muestran conceptos u objetos del dominio del problema y asociaciones entre las clases conceptuales. En la fase de inicio se determinó que los procesos del negocio no están claramente definidos, por esta razón se decide representar la situación real del sistema mediante un Modelo de Dominio, facilitando a través de un vocabulario común que ayude a usuarios, clientes, desarrolladores e interesados a comprender el argumento principal del sistema.

2.1.1 Diagrama de clases del dominio

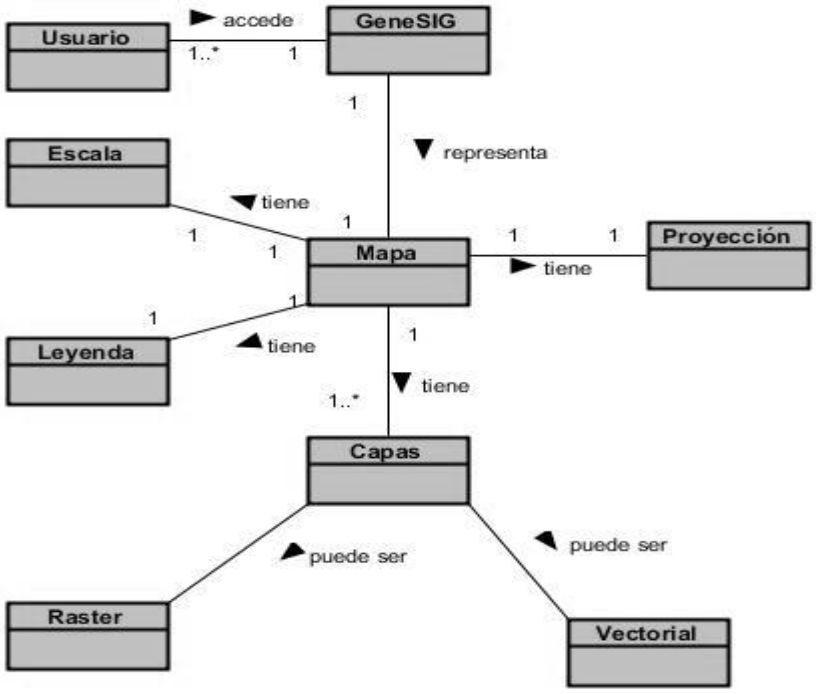

FIG. 4 MODELO DE DOMINIO

2.1.2 Descripción del modelo de dominio

El usuario accede a la plataforma soberana GeneSIG para realizar un análisis del terreno, obteniendo así una representación de un mapa que contiene información en formato *raster*. Un mapa posee una escala, una leyenda, una proyección y un conjunto de capas las cuales pueden ser de tipo *raster* o vectorial, que permiten un mejor entendimiento y análisis de los mismos.

2.1.3 Definición de las clases del modelo de dominio

Usuario: Persona que accede a la plataforma GeneSIG con el objetivo de realizar análisis del terreno a partir de información geográfica almacenada en formato *raster*.

GeneSIG: Plataforma que permite realizar representaciones y análisis de información referenciada geográficamente.

Mapa: Representación gráfica y métrica de una porción de territorio sobre la superficie bidimensional, generalmente plana, pero que puede ser también esférica como ocurre en los globos terráqueos.

Escala: Relación entre la distancia que separa dos puntos en un mapa y la distancia real de esos dos puntos en la superficie terrestre. En los mapas, la escala puede expresarse de tres modos distintos: en forma de proporción o fracción, con una escala gráfica o una expresión en palabras y cifras. Cuanto mayor es la escala, más se aproxima al tamaño real de los elementos de la superficie terrestre.

Proyección: Es un sistema de representación gráfico que establece una relación ordenada entre los puntos de la superficie curva de la Tierra y los de una superficie plana (mapa). Estos puntos se localizan auxiliándose en una red de meridianos y paralelos, en forma de malla.

Capa: Conjunto lógico de elementos temáticos descritos y almacenados en una biblioteca. Estas capas organizan la biblioteca según temas.

Leyenda: Explicación de los símbolos, los colores, las tramas y los sombreados empleados en un mapa.

Modelo de dato vectorial: En los datos vectoriales, el interés de las representaciones se centra en la precisión de localización de los elementos geográficos sobre el espacio y donde los fenómenos a representar son discretos, es decir, de límites definidos. Cada una de estas geometrías está vinculada a una fila en una base de datos que describe sus atributos.

Modelo de dato *raster*: Un tipo de datos *raster* es, en esencia, cualquier tipo de imagen digital representada en mallas. El modelo de SIG *raster* o de retícula se centra en las propiedades del espacio

más que en la precisión de la localización. Divide el espacio en celdas regulares donde cada una de ellas representa un único valor.

2.2 Descripción de la solución

El componente para el análisis del terreno sobre el modelo de presentación *raster* tiene que generar cinco tipos fundamentales de análisis, estos son: mapa de sombra, análisis de pendiente, análisis de orientación, mapa de índice de escabrosidad del terreno y mapa de índice de posición topológica. El usuario podrá especificar los valores y atributos necesarios para cada uno de estos tipos de análisis.

Para realizar el análisis el usuario seleccionará una imagen y el resultado podrá ser visualizado en la plataforma GeneSIG o se guardará una imagen en formato .TIF en una ubicación seleccionada por el propio usuario.

2.3 Requisitos del software.

Los requisitos de software son la descripción de los servicios proporcionados por el sistema y sus restricciones operativas. Se pueden clasificar en funcionales y no funcionales (Sommerville, 2005) y se identifican a partir de las necesidades que tiene el usuario. Existen varias técnicas que facilitan la comunicación con los clientes del producto con el fin de lograr su satisfacción entre las que se encuentran entrevistas, cuestionarios, tormenta de ideas y observación. Sobre la base del análisis de las soluciones existentes en el capítulo anterior, la representación de los conceptos u objetos del mundo real mediante el modelo de dominio y la necesidad de realizar una correcta captura de los requisitos, para evitar demoras en la construcción del componente y errores en los mismos se realizará una tormenta de ideas con especialistas de Aplicativos SIG (Sommerville, 2005).

2.3.1 Requisitos Funcionales.

Los requisitos del sistema son la condición que debe ser alcanzada o poseída por un sistema o componente de un sistema para satisfacer un contrato, estándar u otro documento impuesto formalmente. Se denomina como: condición o capacidad que necesita un usuario para resolver un problema o lograr un objetivo, define qué es lo que el sistema debe hacer, para lo cual se identifican las funcionalidades requeridas y las restricciones que se imponen (Tecnología y Synergix, 2008).

Partiendo de la descripción de las clases más importantes dentro del contexto del sistema representado en el Modelo de Dominio, se determinaron los requisitos funcionales.

RF1. Generar Mapa de Sombra

Descripción: El sistema debe ser capaz de permitir al usuario seleccionar una imagen en formato *raster* y transformarla en un mapa de sombra.

Entrada: imagen en formato *raster*

Salida: mapa de sombra

RF2. Generar Mapa de Sombra utilizando la fórmula de Zevenbargen&Thorne

Descripción: El sistema debe ser capaz de permitir al usuario seleccionar una imagen en formato *raster* y aplicando la fórmula de Zevenbargen&Thorne transformarla en un mapa de sombra

Entrada: imagen en formato *raster* y marcar la opción utilizar la fórmula de Zevenbargen&Thorne.

Salida: mapa de sombra.

RF3. Generar Mapa de Pendiente

Descripción: El sistema debe ser capaz de permitir al usuario seleccionar una imagen en formato *raster* y transformarla en un mapa de pendiente.

Entrada: imagen en formato *raster*.

Salida: mapa de pendiente.

RF4. Generar Mapa de Pendiente utilizando la fórmula de Zevenbargen&Thorne

Descripción: El sistema debe ser capaz de permitir al usuario seleccionar una imagen en formato *raster* y aplicando la fórmula de Zevenbargen&Thorne transformarla en un mapa de pendiente.

Entrada: imagen en formato *raster* y marcar la opción utilizar la fórmula de Zevenbargen&Thorne.

Salida: mapa de pendiente.

RF5. Generar Mapa de Orientación

Descripción: El sistema debe ser capaz de permitir al usuario seleccionar una imagen en formato *raster* y transformarla en mapa de orientación.

Entrada: imagen en formato *raster*

Salida: mapa de orientación.

RF6. Generar Mapa de Orientación utilizando la fórmula de Zevenbargen&Thorne

Descripción: El sistema debe ser capaz de permitir al usuario seleccionar una imagen en formato *raster* y aplicando la fórmula de Zevenbargen&Thorne transformarla en un mapa de orientación.

Entrada: imagen en formato *raster* y marcar la opción utilizar la fórmula de Zevenbargen&Thorne.

Salida: mapa de orientación.

RF7. Generar Mapa de Índice de escabrosidad del terreno

Descripción: El sistema debe ser capaz de permitir al usuario seleccionar una imagen en formato *raster* y transformarla en mapa de índice de oscuridad del terreno.

Entrada: imagen en formato *raster*.

Salida: mapa de índice de oscuridad.

RF8. Generar Mapa de Índice de posición topológica

Descripción: El sistema debe ser capaz de permitir al usuario seleccionar una imagen en formato *raster* y transformarla en mapa de índice de posición topológica.

Entrada: imagen en formato *raster*.

Salida: mapa de índice de oscuridad.

RF9. Mostrar el mapa resultante en GeneSIG

Descripción: El sistema debe ser capaz de permitir al usuario visualizar el mapa (mapa de sombra, mapa de Sombra utilizando la fórmula de Zevenbargen&Thorne, mapa de pendiente, mapa de Pendiente utilizando la fórmula de Zevenbargen&Thorne, mapa de orientación, mapa de Orientación utilizando la fórmula de Zevenbargen&Thorne, mapa de índice de oscuridad del terreno y mapa de índice de posición topológica) en la plataforma GeneSIG.

Entrada: mapa (mapa de Sombra, mapa de Sombra utilizando la fórmula de Zevenbargen&Thorne, mapa de Pendiente, mapa de Pendiente utilizando la fórmula de Zevenbargen&Thorne, mapa de Orientación, mapa de Orientación utilizando la fórmula de Zevenbargen&Thorne, mapa de Índice de oscuridad del terreno y mapa de índice de posición topológica).

Salida: mapa (mapa de sombra, mapa de Sombra utilizando la fórmula de Zevenbargen&Thorne, mapa de pendiente, mapa de Pendiente utilizando la fórmula de Zevenbargen&Thorne, mapa de orientación, mapa de Orientación utilizando la fórmula de Zevenbargen&Thorne, mapa de índice de oscuridad del terreno y mapa de índice de posición topológica) en la plataforma GeneSIG.

2.3.2 Requisitos no funcionales.

Los requisitos no funcionales (RNF) son las propiedades o cualidades que el producto debe tener. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido y confiable. Normalmente están vinculados a requerimientos funcionales, es decir, una vez que se conozca lo que el sistema debe hacer se puede determinar cómo ha de comportarse, qué cualidades debe tener o

cuán rápido o grande debe ser. Los requerimientos no funcionales también añaden funcionalidad al producto, pues hacen que un producto sea fácil de usar, seguro, o demanda cierta cantidad de procesamiento.

RNF1. Apariencia o interfaz externa

Se desea que la interfaz externa del componente sea de fácil navegación por el usuario, sencilla y legible, que mantenga los estándares y características de la plataforma soberana GeneSIG como son: los colores, la estructura de los botones, el estilo y tamaño de letra, ya que el componente a construir va a ser incorporado a la plataforma. Para lograr una interfaz de usuario amigable, atractiva y funcional para el usuario final, es necesario tener en cuenta algunos principios de diseño de interfaz de usuario que serán listados a continuación:

- Las funcionalidades deberán estar al alcance de un clic y representadas por íconos asociados a la acción que se realiza, de manera que cualquier persona con un mínimo dominio de la informática pueda hacer uso del sistema.
- Garantizar la legibilidad de manera que exista contraste de los colores de los textos con el fondo y el tamaño de la fuente sea lo suficientemente adecuado a la vista del usuario.
- Los mensajes mostrados al usuario deben ser concisos y de fácil comprensión.
- Los menús y etiquetas de botones deben comenzar con la palabra más importante.

RNF2. Software

Se requiere para las estaciones de trabajo servidoras las siguientes condiciones:

- Servidor web Apache en su versión 2.0 o superior.
- PostGIS 2.0 como extensión de PostgreSQL, como soporte de datos espaciales.
- Sistema Operativo: GNU/Linux Ubuntu Server 14.04.
- GDAL versión 1.10.1

Se requiere para las estaciones de trabajo cliente las siguientes condiciones:

- Navegador web que cumpla con los estándares W3C (por su siglas en inglés, *World Wide Web Consortium*, es un consorcio internacional que produce recomendaciones para la web mundial).

RNF3. Hardware

Se requiere para las estaciones de trabajo servidoras las siguientes condiciones:

- Debe poseer tarjeta de red.
- Procesador: 3Ghz.

- Memoria RAM: 2Gb.
- Disco Duro: 160Gb.

Se requiere para las estaciones de trabajo cliente las siguientes condiciones:

- Debe poseer tarjeta de red.
- Procesador: 512MHz.
- Memoria RAM: 128Mb.
- Disco Duro: 40Gb.

2.4 Arquitectura del Sistema

La arquitectura del sistema es la organización fundamental de un sistema encarnada en sus componentes, la relación entre ellos y el ambiente y los principios que orientan su diseño y evolución (Kiccillof, 2004).

"La arquitectura del software de un programa o sistema de cómputo es la estructura o estructuras del sistema, lo que comprende a los componentes del software, sus propiedades externas visibles y las relaciones entre ellos". Es decir, es una representación que permite analizar la efectividad del diseño para cumplir los requerimientos establecidos, considerar alternativas arquitectónicas en una etapa en la que hacer cambios al diseño todavía es relativamente fácil y reducir los riesgos asociados con la construcción del software (Pressman, 2010).

2.4.1 Patrones Arquitectonicos

Un patrón arquitectónico impone una transformación en el diseño de la arquitectura, su alcance es más específico ya que se concentra en un aspecto, en lugar de hacerlo en toda la arquitectura. Un patrón aplica una regla sobre la arquitectura, describe la manera en que el software maneja algún aspecto de su funcionalidad al nivel de la infraestructura, abarca aspectos específicos del comportamiento dentro del contexto de la arquitectura y es usado para determinar la forma de la estructura general de un sistema.

El componente Análisis del terreno para la plataforma GeneSIG, se ha desarrollado utilizando la misma arquitectura de la plataforma GeneSIG para minimizar los riesgos en la integración. Los patrones arquitectónicos utilizados son: basado en componente, orientado a objeto y cliente-servidor.

Patrón Arquitectónico Basado en Componente

Este define cómo organizar el modelo en componentes funcionales, exponiendo interfaces de comunicación que contienen métodos, eventos y propiedades. El mismo permite que se pueda

representar el componente como un componente que es incorporado a la plataforma GeneSIG, de manera que no altere los restantes componentes, ni la estructura de la misma; brindando la posibilidad de que el sistema sea flexible y fácil de personalizar.

Patrón Arquitectónico Orientado a Objeto

Se basa en los principios de la Programación Orientada a Objetos (POO): encapsulamiento, herencia y polimorfismo. Sus componentes son los objetos, o más bien instancias de los tipos de dato abstracto. Las interfaces están separadas de las implementaciones y se puede modificar la implementación de un objeto sin afectar a sus clientes (Camacho, 2004).

Estilo Cliente-Servidor

La arquitectura Cliente-Servidor definido como un modelo organiza un sistema como un conjunto de servicios y servidores asociados, más unos clientes que acceden y usan los servicios. Los principales componentes de estos servicios son (Sommerville, 2005):

1. Un conjunto se servidores que ofrecen servicios a otros sistemas.
2. Un conjunto de clientes que llaman a los servicios ofrecidos por los servidores.
3. Una red que permite a los clientes acceder a los servicios.

De manera más informal la arquitectura Cliente-Servidor se identifica como un componente servidor, que ofrece ciertos servicios, escucha que algún otro componente requiera uno; un componente cliente solicita ese servicio al servidor a través de un conector. El servidor ejecuta el requerimiento (o lo rechaza) y devuelve una respuesta (Reynoso, 2004). La principal ventaja de este modelo es que pueden insertarse clientes y servidores sin afectar la distribución del sistema.

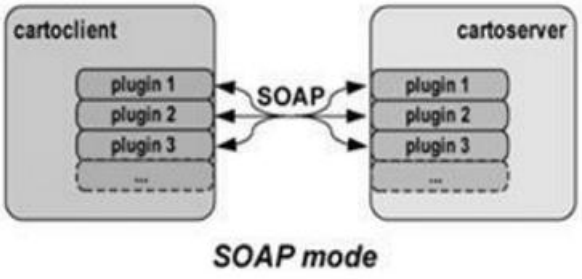

FIG. 5 SOAP

El modo **SOAP** como su nombre indica, plantea que dicha comunicación debe ser a través del protocolo SOAP, en este caso la comunicación entre CartoClient y el CartoServerserá mediante servicios web.

2.4.2 Patrones de Diseño

En el diseño de la propuesta de solución se aplican los patrones GRASP, el cual es un acrónimo que significa *General Responsibility Asignment Software Patterns* (patrones generales de software para asignar responsabilidades). A continuación se mencionan los patrones GRASP utilizados, que permiten describir los principios fundamentales de diseño de objetos para la asignación de responsabilidades.

Experto: Se aplica para asignar una responsabilidad al experto en información: la clase que cuenta con la información necesaria para cumplir la responsabilidad. En este caso, se tiene la clase ClientModelo, que es experta en procesar los datos que son enviados a través del navegador web, la clase ServerModelo, que es la responsable de la interacción con el servidor de mapas y el servidor de bases de datos y la clase Modelo.ajax, experta en el comportamiento por la parte cliente del *plugin*. El uso de este patrón permite que se conserve el encapsulamiento, ya que los objetos se valen de su propia información para hacer lo que se les pide. Esto soporta un bajo acoplamiento, lo que favorece al hecho de tener sistemas más robustos y de fácil mantenimiento.

Creador: Se aplica para la asignación de responsabilidades a las clases relacionadas con la creación de objetos, de forma tal que una instancia de un objeto sólo pueda ser creada por el objeto que contiene la información necesaria para ello. En este caso, el patrón se refleja en las clases ClientModelo y ServerModelo, encargadas de crear una instancia de las clases ModeloRequest y ModeloResult, clases que describen las variables que contienen la información que forman parte de los valores de entrada de la solicitud que realiza el ClientModelo al ServerModelo y viceversa.

Bajo Acoplamiento: El acoplamiento es una medida de la fuerza en que una clase está conectada a otras clases, que las conoce y recurre a ellas. En este caso, se refleja el bajo acoplamiento, en cada una de las clases del componente Análisis del Terreno, con el objetivo de que una clase no dependa de muchas clases, de esta forma, no se afectan las clases por cambios de otros componentes, son fáciles de entender por separado y fáciles de reutilizar.

Alta Cohesión: La cohesión es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. En este caso, se garantiza que cada una de las clases del componente Análisis del Terreno, posean alta cohesión, de manera que las clases posean la característica de tener las responsabilidades estrechamente relacionadas y que no realicen un trabajo enorme. El uso de este patrón permite que se pueda mejorar la claridad y facilidad en que se entiende el diseño, se simplifique el mantenimiento y existan mejoras de funcionalidad.

Controlador: Un controlador es un objeto de interfaz no destinada al usuario que se encarga de manejar un evento del sistema. Define además el método de su operación. En este caso, se encuentra reflejado en la clase Modelo.ajax, encargada del comportamiento del *plugin* y gestiona todos los eventos que ocurren en el mismo.

Command: este patrón permite encapsular las peticiones a través de un objeto, lo que permite realizar operaciones como gestionar las acciones de dicho objeto. Se utiliza para la comunicación a través de las interfaces de usuario, específicamente a través de la clase **AJAXHelper** que es la encargada de comunicar las interfaces con el servidor. Uno de los aspectos más importantes en el sistema son las interfaces gráficas de usuario, ya que el usuario interactúa constantemente con ellas y por eso principalmente se aplica este patrón GoF a la solución.

2.5 Diagrama de Clase del Diseño

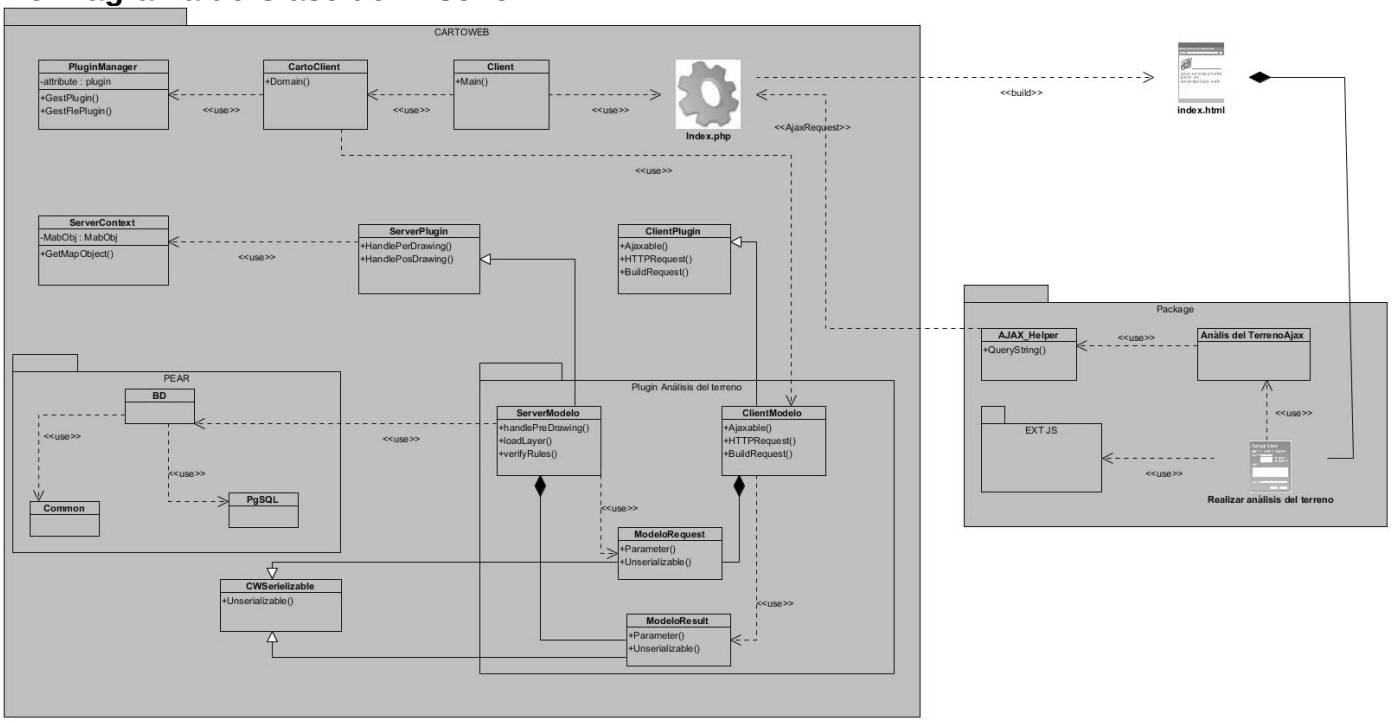

FIG. 6 MODELO DE DISEÑO

2.5.1 Descripción de las clases del diseño

- **index.php:** tiene como propósito controlar la realización del RF en sí, recibe las peticiones realizadas por el cliente, gestiona las mismas y manda a construir la ClientPage.
- **Client:** contiene todos los archivos específicos de PHP del lado de CartoClient y permite la interacción entre la index.php y la CartoClient.
- **CartoClient:** integra y recoge todos los datos y funciones realizadas por cada una de las .js que intervienen en el RF y se definen una serie de variables globales que van a ser utilizadas por la aplicación.
- **PluginManager:** clase que se utiliza para gestionar la base de *plugins*.
- **ClientPlugin:** contiene las interfaces necesarias para los *plugins* del lado del cliente.
- **ServerPlugin:** esta clase proporciona la base de herramientas para el desarrollo de *plugins*.
- **BD:** es la clase encargada de establecer la conexión con el servidor de base de datos para procesar los objetos a editar.
- **ServerContex:** es la contenedora de la información común que ha de ser utilizada por la parte cliente y la servidora, empleando la información seleccionada como un objeto para un fácil manejo de los datos.
- **Common:** es la encargada de administrar las conexiones a la base de datos para ejecutar las consultas a la misma satisfactoriamente, esto incluye tratamiento de los datos.
- **Pgsql:** gestiona desde PHP las funciones de PostgreSQL.
- **CWSerializable:** se encarga de serializar todas aquellas clases que pueden ser serializadas, permitiendo la comunicación entre el Client y el Server del *plugin*.
- **AJAX_Helper:** tiene como propósito enviar las respuestas de los *plugins* “AJAX”, para alimentar a los *plugins* que responden a las peticiones del usuario.
- **Análsis del TerrenoAjax:** es la encargada de gestionar el pedido y respuesta a las peticiones del usuario por Ajax.
- **ModeloRequest:** es una clase común encargada de transportar los datos recogidos en **ClientModelo** desde la interfaz y transportarlos a la clase **ServerModelo**.

- **ModeloResult**: es una clase común encargada de transportar los datos generados en **ServerModelo** a la clase **ClientModelo**.
- **ClientModelo:** se encarga de recoger y seleccionar de las .js contenidas en el paquete JS, toda la información correspondiente a los atributos y valores para procesar la imagen, entrados a través de los formularios, y los envía al **ServerModelo**.
- **ServerModelo:** es la clase servidora que se encarga de procesar los datos, generar la imagen *raster* y enviar las respuestas necesarias al **ClientModelo**.
- **Index.html:** es la encargada de mostrar en el mapa la región localizada.
- **Realizar Análisis del Terreno:** Tiene como objetivo permitir al usuario introducir todos los datos y atributos de la imagen que se desea procesar.

2.6 Conclusiones

La confección del diagrama de clases del dominio permite visualizar la relación existente entre los conceptos u objetos significativos en el dominio del problema. Con dicho artefacto, la obtención de los requisitos funcionales y no funcionales, de conjunto con el modelado del sistema, proporciona una mejor comprensión para el equipo de desarrollo. La utilización de patrones de diseño en la implementación de la solución propicia un ahorro de tiempo en el desarrollo. Con el uso de la metodología Prodesoft se construye un software con un diseño simple pero adecuado.

Introducción

En el presente capítulo de las disciplinas implementación y prueba se realizará una descripción de cómo los elementos del modelo de diseño se implementan en términos de componentes .Se llevarán a cabo las pruebas para validar el componente para el análisis del terreno en la Plataforma GeneSIG. Además, se estarán generando las pruebas aplicadas a la solución informática mediante el método de prueba, caja negra, utilizando la técnica de partición equivalente, así como los resultados que demuestran el grado de satisfacción de la aplicación informática desarrollada.

3.1 Estilo de programación

Los estilos de programación definen la estructura y apariencia física del código, lo que facilita su comprensión, mantenimiento y lectura. En la implementación se utilizaron diferentes estilos que se describen a continuación:

3.1.1 Definición de clases

Las clases poseen la siguiente estructura siguiendo un estilo CamelCase.

```php
<?php
class ClientModelo extends ClientPlugin implements GuiProvider, ServerCaller, Ajaxable
){
    public $action;
    public $result;
    public $result1;
    public $datos;
    public $msg;
```

FIG. 7 DEFINICIÓN DE CLASES

3.1.2 Estilo de métodos

Los métodos tendrán letra inicial minúscula, en caso de ser dos palabras se mantendrán unidas y la segunda comenzará con mayúscula. En caso de pasarle elementos por parámetros, estos se escribirán con minúscula.

```
createLayer: function(params){
 if(params!='undefined')
 {
  Genesig.addvectorLayer(params.id, params.nameLayer, params.attributes, params.register);
 }
},
crearCapa: function(){
 var layer={
            nameLayer:'Generada',
            id:"generadacapa",
            attributes:{
            projection:AjaxPlugins.StaticTools.getProjectionSRSCode(),
            overflowLayer:true,
            icon:'ico_layer_cosmetic'
            }
           };
 this.createLayer(layer);
 this.ejemplo=Genesig.getLayer("generadacapa");
}
```

FIG. 8 ESTILO DE MÉTODO

3.1.3 Declaración de variables

Las variables en javascript se escribirán con minúscula y estarán antecedidas de la palabra reservada **var**. El signo igual (=) estará ubicado entre espacios.

```
var objView = AjaxPlugins.Modelo.View;
var result = Ext.decode(pluginOutput.variables.result);
var result1 = Ext.decode(pluginOutput.variables.result1);
var result2 = Ext.decode(pluginOutput.variables.result2);
var msg = Ext.decode(pluginOutput.variables.msg);
```

FIG. 9 DECLARACIÓN DE VARIABLE

3.2 Diagrama de componente

Un componente es una unidad modular que puede reemplazarse en su propio entorno y cuyos elementos internos quedan ocultos. Puede poseer interfaces proporcionadas, a través de las cuales se puede obtener acceso a sus funciones, e interfaces necesarias, en las que se definen las funciones o servicios de otros componentes que son necesarios. Mediante la conexión de las interfaces proporcionadas y las interfaces necesarias de distintos componentes puede construirse un componente mayor. Un sistema de software completo se puede concebir como un componente (Jacobson, 2000).

El diagrama de componentes permite concebir el diseño atendiendo a los bloques principales y ayuda al equipo de desarrollo a entender un diseño existente y crear uno nuevo. Al establecer el sistema como una colección de componentes con interfaces proporcionadas y necesarias bien definidas se garantiza la correcta separación entre los componentes. A su vez, se facilita la comprensión de los cambios al modificar los requisitos. Este tipo de diagrama representa la parte modular, desplegable y reemplazable de un sistema que encapsula implementación. También expone un conjunto de interfaces que funcionan como elementos intermediarios de comunicación entre componentes que posibilitan la independencia funcional (Jacobson, 2000). A continuación se propone el diagrama de componentes para el componente “Análisis del terreno para la plataforma GeneSig”, representando de forma física el sistema.

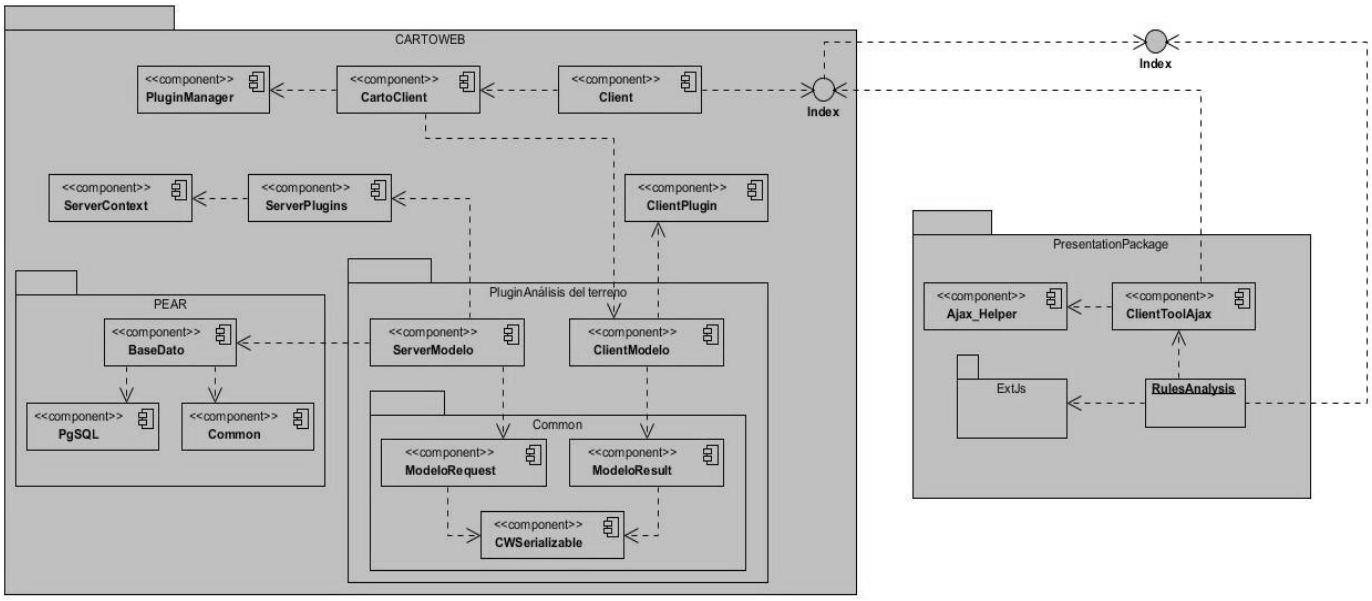

FIG. 10 DIAGRAMA DE COMPONENTE

3.3 Pruebas

Las pruebas representan actividades claves que ayudan a entregar el producto con la calidad requerida para satisfacer las necesidades del cliente, cumplir con sus expectativas y con la certeza de que el producto cumple las especificaciones definidas.

Concretamente la prueba de software se puede definir como una actividad en la cual un sistema o uno de sus componentes se ejecutan en circunstancias previamente especificadas, registrándose los resultados obtenidos (Juristo, 2005). Seguidamente se realiza un proceso de evaluación en el que los resultados obtenidos se comparan con los resultados esperados para localizar fallos en el software. Las técnicas de evaluación dinámica o prueba proporcionan distintos criterios para generar casos de prueba que provoquen fallos en los programas. Estas técnicas se agrupan en Técnicas de Caja Blanca o Estructurales, que se basan en un minucioso examen de los detalles procedimentales del código a evaluar; y Técnicas de Caja Negra o Funcionales, que realizan pruebas sobre la interfaz del programa a probar, entendiendo por interfaz las entradas y salidas de dicho programa. Estas son técnicas complementarias que han de aplicarse al realizar una prueba dinámica, ya que pueden ayudar a identificar distintos tipos de fallas en un programa.

Un diseño de caso de prueba (DCP) está compuesto por un conjunto de entradas, respuesta que emite el sistema de acuerdo a esas entradas y el flujo central que indica el camino del escenario descrito. Estos son desarrollados para verificar el cumplimiento total o parcial de un requisito. Las entradas representan las variables que se pueden especificar y las mismas contienen: V, I, o N/A. V indica válido, I indica inválido, y N/A que no es necesario proporcionar un valor del dato en este caso, ya que es irrelevante. A continuación se presenta en la Tabla #1, un ejemplo de los DCP realizados para comprobar el

funcionamiento de la solución, específicamente para el requisito funcional número 1. Realizar mapa de sombra:

TABLA 1 CASO DE PRUEBA MAPA DE SOMBRA

Nombre de la sección	Escenarios de la sección	Archivo de entrada	Banda	Factor		Escala	Azimut de la luz	Altitud de la luz	Respuesta del Sistema	Flujo Central
SC1: Realizar mapa de sombra	EC1.1: Realizar mapa de sombra correctamente.	V	V	V		V	V	V	Muestra el mapa de sombra resultante.	El usuario selecciona una imagen de entrada, después selecciona la banda deseada o simplemente la deja nula, después selecciona el modo mapa de sombra. A continuación introduce los valores para: Factor, escala, azimut de la luz y altitud de la luz, y selecciona la opción de los checkbox deseados, en caso de seleccionar opciones de creación debe seleccionar el perfil deseado. Por último se da clic al botón aceptar.
	EC1.2: Campos Obligatorios vacíos.	I	V	V		V	V	V	El sistema notifica que hay campos vacíos	El usuario deja el archivo de entrada vacío.
			N/A	5		3	450	35		
	EC1.3: Formato incorrecto para el archivo de entrada	I	V	V		V	V	V	El sistema notifica que el formato del archivo de entrada no es correcto	El usuario introduce en el archivo de entrada un objeto no valido.
		Empresa.avi	1	2		3	400	43		
	EC1.4: Cancelar Ventana	V	V	V		V	V	V	El sistema cancela la ventana y muestra la interfaz principal de GeneSIG	El usuario le da al botón Cancelar
		N/A	N/A	1		1	315	45		
	EC1.5: Cerrar	V	V	V		V	V	V	El sistema	El usuario da clic en el botón

	Ventana.	N/A	N/A	1		1	315	45	oculta la ventana.	cerrar

3.3.1 Resultados de las pruebas realizadas

Fueron detectadas un total de 10 no conformidades durante 3 iteraciones, las cuales fueron corregidas a medida que se fue avanzando en el proceso de prueba. A continuación se muestra en la siguiente ilustración las No conformidades detectadas y su clasificación:

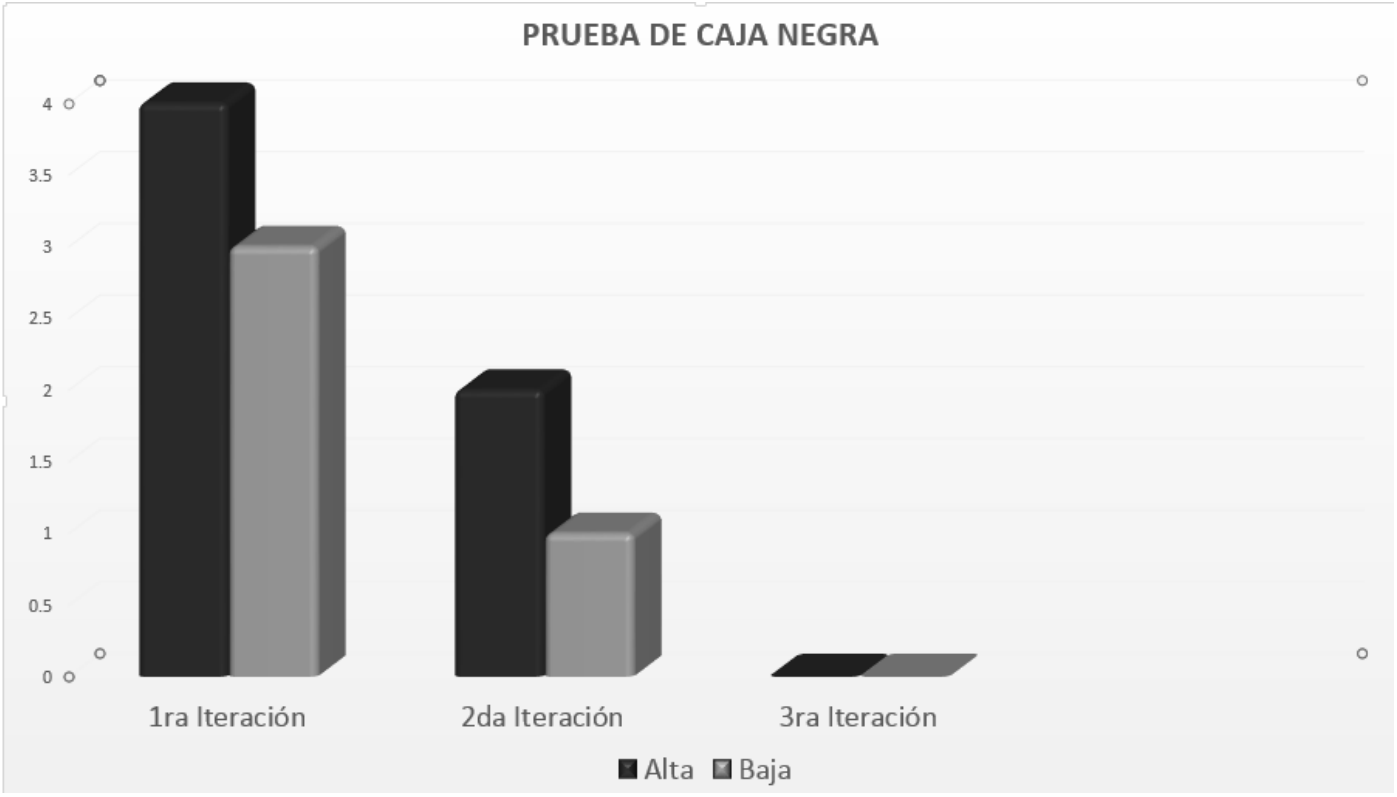

FIG. 11 PRUEBA DE CAJA NEGRA

A continuación en el "**Registro de defectos y dificultades detectados"** se muestran ejemplos de las no conformidades detectadas durante las pruebas a la aplicación y la respuesta del equipo de desarrollo:

TABLA 2 REGISTRO DE DEFECTOS Y DIFICULTADES DETECTADOS

Elemento	No.	No conformidad (NC)	Alta	Baja	Resp. equipo de desarrollo
Aplicación	1	Al dejar el campo de archivo de entrada vacío el sistema no valida que el campo se encuentre vacío o no	X		En la clase ModeloDynamic.js se validó que el archivo de entrada no esté vacío.
Aplicación	2	Al dar clic en el botón cancelar la aplicación no ejecuta ninguna acción.		X	Se implementó que al dar clic en el botón cancelar la ventana se cerrara.
Aplicación	3	Al generar un mapa de sombra la aplicación dio un error inesperado y no informó al usuario del mismo.	X		Se verificó que el error fue porque no estaba instalada la librería GDAL y se validó en la clase ServerModelo.php que antes de realizar la operación la librería estuviera instalada de no ser así se le informaría al usuario.
Aplicación	4	En la interfaz principal la palabra creación no tenía tilde		X	Se corrigió la palabra poniéndole la tilde
Aplicación	5	Las opciones de creación aparecen activadas por defecto y deberían		X	En la clase ModeloDynamic.js se configuró

		aparecer desactivadas.			para que aparezca desactivado por defecto el checkbox opciones de creación
Aplicación	6	La aplicación aunque marques o desmarques la opción visualizar mapa este muestra de cualquier manera	X		En la clase ServerModelo.php se validó mostrar solo el mapa cuando el checkbox visualizar mapa esté seleccionado
Aplicación	7	Cuando se genera un mapa de pendiente y se deja marcada la opción mostrar en el mapa este no se muestra.	X		Se corrigió en la clase ServerModelo.php los atributos del mapa de pendiente que estaban escrito de forma incorrecta
Aplicación	8	Cuando se genera un segundo mapa de sombra el primero se pierde	X		En la clase ServerModelo.php se le puso un nombre genérico al archivo de salida para que no se sobreescribiera
Aplicación	9	El atributo del mapa de sombra azimut de la luz está escrito incorrectamente		X	Se corrigió la palabra poniéndole la tilde
Aplicación	10	El sistema no valida que los objetos de creación sean correctas	X		En la clase ModeloDynamic.js se validaron que las opciones de creación sean las permitidas por la librería GDAL.

3.4 Conclusiones

El estilo de programación utilizado en la implementación del componente es indispensable y posibilita un mejor entendimiento y organización del código. Una vez terminado el componente se puede apreciar que la interfaz tiene un diseño simple que posibilita una fácil interacción con el usuario. Por otro lado los casos de pruebas son adecuados, pues garantiza que la aplicación funcione correctamente teniendo en cuenta lo que se espera de esta.

Conclusiones Generales

Una vez culminada la investigación es posible afirmar que se dio cumplimiento al objetivo general trazado, por lo que se concluye que:

- El estudio de las aplicaciones existentes a nivel internacional aunque no responden totalmente al problema, sirvieron de base para elaborar una idea general de la forma en que se realiza el análisis del terreno, en los sistemas SIG y las ventajas que proporciona el desarrollo de un componente que permita ejecutar dicho proceso.
- La elección de Prodesoft como proceso de desarrollo de software permitió guiar y documentar el ciclo de vida de la aplicación, permitiendo que esta cumpliera con los estándares y estilos definidos en las líneas bases de GeneSIG.
- La utilización de las tecnologías y herramientas definidas en las líneas bases de GeneSIG permitió que la solución se integre a esta plataforma sin provocar incompatibilidades, además estas obedecen a criterios de selección de tecnologías libres garantizando las políticas que impulsa la universidad y el país.
- La realización de las pruebas de caja negra permitió detectar los posibles errores, para así dar cumplimiento a los requisitos funcionales y no funcionales establecidos y obtener un producto completamente funcional.
- Se cumplió el objetivo planteado logrando desarrollar un componente para la plataforma GeneSIG que permite facilitar el análisis de las características del terreno a partir de la información geográfica almacenada en formato *raster*.

Referencias Bibliográficas

Asociacion gvSIG. 2009. [En línea] 2009. [Citado el: 2 20, 2016.] http://www.gvsig.com/productos/gvsig-desktop.

Bardales, Erik Arnulfo Santizo. 2007. ANÁLISIS PROGRAMACIÓN DE COMPUTADORAS. [En línea] 2007. [Citado el: 1 14, 2016.] http://biblioteca.usac.edu.gt/tesis/08/08_0341_CS.pdf.

Bravo, Javier Dominguez. 2000. *"Breve Introducción a la Cartografía".* Madrid : s.n., 2000. 1135-9420.

Camacho, Erika, Cardeso, Fabio y Núñez,. 2004. *Gabriel. Arquitecturas de Software, Guía de estudio.* 2004.

Colectivo de autores. 2002. *Ciencia, Sociedad y Tecnología.* 2002. 959-05-0283-0.

Dave W. Mercer, By (author) Allan Kent , Steven D. Nowicki , Dan Squier , Wankyu Choi. 2015. Fundamentos PHP 5/ Beginning PHP 5. [En línea] 2015. [Citado el: 1 3, 2016.] /www.bookdepository.com/Fundamentos-PHP-5-Beginning-PHP-5-Dave-Mercer/9788441518056.

de Cos Guerra. 2012. Posibilidades de los SIG en el tratamiento de la cartografía digita. [En línea] octubre 2012. [Citado el: 2 8, 2016.] http://www.rcg.cat/articles.php?id=247.

—. 2012. Posibilidades de los SIG en el tratamiento de la cartografía digital. [En línea] octubre 2012. [Citado el: 2 10, 2016.] http://www.rcg.cat/articles.php?id=247.

Delavaut, Dr. C. Raúl Rubén Fernández Aedo y Lic. Martín Enrique. 2012. *Educacion y tecnologia, un binomio excepcional.* s.l. : Grupo editor K, 2012.

DROZDEK, ADAM. ESTRUCTURA DE DATOS ALGORITMO JAVA . [En línea] [Citado el: 1 4, 2016.] http://www.casadellibro.com/libro-estructura-de-datos-algoritmo-java/9789706866110/1145636.

Ecured. Case. [En línea] [Citado el: 1 11, 2016.] http://www.ecured.cu/CASE.

Ecured. Desarrollo de Software. [En línea] [Citado el: 1 6, 2016.] http://www.ecured.cu/Desarrollo_de_software.

Ecured. UML. [En línea] [Citado el: 1 11, 2016.] http://www.ecured.cu/index.php/UML.

ENRI. 1995. [En línea] 1995. [Citado el: 2 17, 2016.] http://help.arcgis.com/es/arcgisdesktop/10.0/help/index.html#/na/00r90000001n000000/. 92373-8100.

Felicísimo,A.M. 1994. [En línea] 1994. [Citado el: 10 5, 2015.] http://www.um.es/geograf.

Frederick. 2008. *Learning Ext JS.* UK: Packt Publishing Ltd : s.n., 2008. ISBN 978-1-847195-14-2.

GeneSig, Manual de. 2012. *Manual de GeneSig.* 2012.

Referencias Bibliográficas

GIS Modeling in Raster. 2002. El formato raster.Fundamento. [En línea] 2002. [Citado el: 2 4, 2016.] http://www.um.es/geograf/sigmur/sigpdf/temario_4.pdf.

INSTITUTO TECNOLÓGICO DE VERACRUZ. 2011. INSTITUTO TECNOLÓGICO DE VERACRUZ. [En línea] 2011. [Citado el: NOVIEMBRE 8, 2015.] http://www.prograweb.com.mx/pweb/0301IntroJavaScript.php.

Jacobson. 2000. *El Proceso Unificado de Desarrollo de Software.* Madrid : s.n., 2000.

Juristo. 2005. *Técnicas de evaluación de software.* 2005.

Kiccillof, Carlos Reynoso and Nicolás. 2004. *Estilos y Patrones en la Estrategia de Arquitectura de Microsoft.* 2004.

MappingGIS. 2015. MappingGIS. [En línea] diciembre 17, 2015. [Citado el: 2 26, 2016.] http://mappinggis.com/2014/03/como-trabajar-con-raster-en-qgis/.

Medoly Y. Ivory-Ndiaye. 2016. Software Design Tools for Agile Teams. [En línea] 2016. [Citado el: 4 25, 2016.] http://www.visual-paradim.com/learning.

Méndez, Gonzalo. 2011. Proceso Software y Ciclo de vida. [En línea] 2011. [Citado el: 1 6, 2016.] www.fdi.ucm.es/profesor/gmendez/docs/is0809/02-ProcesoCicloDeVida.pdf.

NetBeans Platform. 2012. NetBeans Platform. [En línea] 2012. [Citado el: 1 4, 2016.] netbeans.org/features/platform/all-docs.html.

Núñez, MSc. Héctor Manuel Fernández. SIG-ESAC: Sistema de Información Geográfica para la gestión de la estadística de salud de Cuba . [En línea] [Citado el: 12 26, 2015.] http://bvs.sld.cu/revistas/hie/vol44_3_06/hie03306.htm#cargo.

Oracle Corporation. 2013. [En línea] 2013. http://netbeans.org/community/releases/70.

PostGIS 2.0 Manual. 2010. [En línea] http://postgis.net/docs/manual-2.0/.

PostgreeSQL. 2013. [En línea] 2013. http://www.postgresql.org.es/sobre_postgresql.

Pressman, Roger S. 2010. *Ingenieria del Sofware.* México : s.n., 2010. 978-607-15-0314-5.

Real Academia Española. 2016. Real Academia Española. [En línea] 2016. http://www.rae.es.

Reynoso, Carlos Billy. 2004. *Introducción a la Arquitectura de Software.* Argentina : Universidad de Buenos Aires., 2004.

2010. Sistema de Información Geográfica, tipo y apliciones empresariales. [En línea] 2010. [Citado el: 10 18, 2015.] http://sig.cea.es/SIG.

Referencias Bibliográficas

Sommerville. 2005. *Ingenieria de software.* Madrid : PEARSON EDUCACIÓN, 2005. ISBN 84-7829-074-5.

Sommerville, Ian. 2005. *Ingeniería de Software.* Madrid : Pearson Educación S.A, 2005. ISBN: 84-7829-074-5.

Taguas, Encarnación. 2010. Predicción de cárcavas efímeras mediante el índice topográfico combinado en una microcuenca de olivar en Andalucía (España). [En línea] 2010. [Citado el: febrero 2, 2016.] http://www.scielo.org.mx/scielo.php?script=sci_arttext&pid=S1405-31952010000400002.

Tecnología y Synergix. 2008. Tecnología y Synergix. [En línea] julio 7, 2008. [Citado el: diciembre 18, 2015.] https://synergix.wordpress.com/2008/07/07/requisito-funcional-y-no-funcional/.

Tutorial de Visual Paradigm for UML. [En línea] [Citado el: 1 11, 2016.] http://es.slideshare.net/fabiannazar1/uml-tutorialvisualparadigm.

Warmerdam. 2008. *The geospatial data abstraction library.* 2008.

Zayas. 1997. 1997.