Universidad de Cienfuegos “Carlos Rafael Rodríguez”

Facultad de Ingeniería

Carrera de Ingeniería Informática

Título: Sistema de Información Geográfica del Centro Meteorológico Provincial de Cienfuegos

Trabajo de Diploma para optar por el título de

Ingeniero Informático

Autor: Marylien Castiñeira Díaz.

Tutor: Ing. Lester Arturo Oropesa Morales.

Co-tutor: Ing. Nidia Dayana Risell Terry.

Cienfuegos, Junio 2014

DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor del trabajo de diploma titulado: “Sistema de Información Geográfica del Centro Meteorológico Provincial de Cienfuegos.” y autorizo a la Facultad de Ingeniería en la Universidad de Cienfuegos “Carlos Rafael Rodríguez”, para que hagan el uso que estimen pertinente del presente trabajo de diploma.

.

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

_____ _____

Firma del Autor Firma del Tutor

PENSAMIENTO

"Con los nuevos descubrimientos de cada día, el hombre tendrá el futuro más brillante que haya soñado jamás."

Frederick L. Schuman

AGRADECIMIENTOS

A mi mamá y a mi papito, que sin ellos no estaría donde estoy, a ellos que nunca dudaron de mi desempeño y me enseñaron que lo que uno se proponga lo puede lograr. Ellos que siempre han estado pendientes de mí, cuidándome y haciéndome ver que si se puede.

A mi tutor Lester por el apoyo y la paciencia.

A mi cotutora Dayana, una verdadera amiga a la cual quiero mucho.

A mis tías Idania, Rebe, Mireya,Vivian, Milagro, que siempre han esperado lo mejor de mí.

A mi tío Fabio.

A mis primas y primos que los quiero mucho y son como hermanos para mí.

A mis hermanas y a mi hermano que siempre los tengo presentes.

A todas mis amistades, que siempre se han preocupado por mí y me han hecho ver que no estoy sola.

A mi abuelo Santo que siempre quiso lo mejor para mí.

A mi abuela Matilde y a mi abuelo Cuco que hoy estarían muy orgullosos de mí.

A RR , muchas gracias por ayudarme.

A Mayuli que espero se recupere pronto.

DEDICATORIA

Por ser lo que más quiero en mi vida, por ser la luz que ilumina mis pasos, la razón de mi existir, por tu paciencia y dedicación, te dedico este trabajo a ti ***mami****.*

RESUMEN

El presente trabajo lleva por título “Sistema de Información Geográfica del Centro Meteorológico Provincial de Cienfuegos”. El desarrollo de este producto informático tuvo como objetivo la configuración de un servidor de mapas vía Web (MapServer), la creación de una base de datos espacial con Postgree/Postgis, la instalación de las herramientas necesarias para analizar y visualizar los datos desde una aplicación Web.

En el documento se describieron los elementos que conforman el análisis, diseño e implementación del sistema propuesto siguiendo lo establecido por la metodología Open UP para la planificación del proyecto, el Lenguaje Unificado de Modelado (UML), para la implementación del mismo y se utilizó JavaScript como lenguaje de programación con el framework Extjs.

Los resultados obtenidos demuestran la viabilidad, pertinencia y actualidad científico técnica de los especialistas en el centro siendo el mayor beneficiado la sociedad pues es el objeto para el cuál ellos laboran. Los resultados obtenidos son validados por el método de especialistas.

PALABRAS CLAVE

Infraestructura, datos espaciales, servidor, mapas, meteorología, software.

Índice

AGRADECIMIENTOS I

DEDICATORIA II

RESUMEN III

INTRODUCCIÓN 4

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA 8

Introducción 8

1.1 Características de los sistemas de Información geográficos. 8

1.1.1 Tipos de datos de un SIG 8

1.1.2 Funcionamiento de los SIG 9

1.1.3 Estándares WEB que garantizan la interoperabilidad. 9

1.1.4 Servicios WMS y WFS 10

1.2 Sistemas de información geográficos aplicados a la meteorología y la climatología. 11

1.3 Proceso de gestión de la información geográfica en el CMP de Cienfuegos. 11

1.4 Metodologías para desarrollo de software 12

1.5 Arquitectura de Software 16

1.6 Tecnologías y herramientas a emplear 22

1.6.1 Sistema de gestor de bases de datos geográficos PostgreSQL/PostGIS. 22

1.6.2 MapServer servidor de mapas en la WEB. 22

1.6.3 Open Layer 22

1.6.4 UML 2.0 23

1.6.5 Herramienta CASE 23

1.6.6 Marco de desarrollo para la aplicación web 24

1.6.7 NetBeans IDE 7.1. 25

Conclusiones del Capítulo 26

CAPÍTULO 2: PLANIFICACIÓN Y CONTROL PARA EL DESARROLLO DEL SOFTWARE 27

Introducción 27

2.1. Normalización de archivo hurdat2 28

2.2. Requisitos no funcionales 30

2.3. Requisitos funcionales 31

2.4. Modelo de Casos de Uso del Sistema (CUS) 32

2.4.1. Definición de los casos de usos del sistema. 32

2.4.2. Diagrama de casos de uso del sistema 33

2.4.3. Patrones de casos de uso utilizados 35

2.5. Diagramas de Clases del Diseño 35

2.6. Diagrama de Secuencia 36

2.7. Patrones arquitectónicos utilizados 37

2.8. Patrones de diseño utilizados 38

2.9. Modelo de Despliegue 38

Conclusiones del Capítulo 39

CAPÍTULO 3: CONSTRUCCIÓN Y VALIDACIÓN 40

Introducción 40

3.1. Diagramas de Componentes 40

3.2. Estándares de codificación 42

3.3. Validación de la Infraestructura mediante el criterio de especialista 43

3.4. Validación de la Infraestructura que se propone por los especialistas 44

Conclusiones del Capítulo 46

CONCLUSIONES GENERALES 47

RECOMENDACIONES 48

REFERENCIAS BIBLIOGRÁFICAS 49

BIBLIOGRAFÍA 51

ANEXOS 53

GLOSARIO DE TÉRMINOS 60

INTRODUCCIÓN

El acelerado desarrollo de la ciencia y la técnica ha irrumpido en todos los ámbitos y niveles sociales, las nuevas tecnologías se han convertido en uno de los productos de mayor consumo en la actualidad [1]. De ellas dependen en gran escala el desarrollo de las sociedades y la economía de los países en el presente siglo.

Muchas empresas hacen uso de las tecnologías con el objetivo de automatizar los procesos que realizan, buscando un aumento de la productividad para introducirse en la competitividad del mercado internacional.

Como consecuencia de este proceso tecnológico evolutivo, surgen las aplicaciones Web, conocidas por representar páginas con contenidos dinámicos que permiten al consumidor mayor interacción. Dentro de ellas están las que se encargan de recibir, guardar y analizar datos. Estas aplicaciones son accedidas desde navegadores Web y deben su popularidad a la independencia del sistema operativo, a lo cómodo y práctico del uso del navegador y que no se distribuyen o instalan en las máquinas clientes [2].

Las Infraestructuras de Datos Espaciales (IDE) corresponden a un concepto que se aplica en diversas escalas en todo el planeta, desde la Infraestructura de Datos espaciales Global (por sus siglas en inglés G.S.D.I, Global Spatial Data Infrastructure), hasta las de orden regional, nacional, estatal o provincial, municipal, local e institucional. Algunas regiones se caracterizan por tener amplia disponibilidad de información geográfica. No obstante, incluso en la impresionante era de las computadoras y la gran red, los usuarios tienen dificultades tanto para encontrar como para utilizar información geográfica determinante. Esto puede conducir al abandono de proyectos o a la repetición innecesaria y costosa de información que ya existe en algún lugar.

En el Centro Meteorológico Provincial de Cienfuegos (CMP) en particular en el Departamento de Pronóstico y Climatológico son insuficientes las herramientas para visualizar y analizar datos espaciales, algunos creados en la institución y otros archivados desde fuentes externas. Es necesario entonces que dicha información esté disponible en la red local y pueda ser consultada por los especialistas del grupo de Pronóstico y Climatológico.

El grupo de clima necesita acceder a los datos para la elaboración de los mapas y realizar las guías climáticas. Dichos datos y fuentes de datos son utilizados para elaborar programas

de impacto realizados en la televisión y/o realizar algún pronóstico diario. También necesitan consultar servicios que estén configurados en otros servidores externos a la red local, que pudieran ser imágenes de satélite en tiempo real o de algún organismo ciclónico de interés en nuestra cuenca geográfica o el atlántico.

En estas condiciones se plantea como **problema científico**: ¿Cómo mejorar la gestión de la información de los datos espaciales que se encuentran disponibles en distintas fuentes de datos y servicios en el CMP?

A partir del problema planteado se define como **idea a defender:** La utilización de un sistema de información geográfica web mejorará la gestión de la información de los datos espaciales en el CMP.

El **objeto de estudio** del presente trabajo son los sistemas de información geográfica.

Como **campo de acción**: Los sistemas de información geográfica aplicados a la climatología y la meteorología.

El trabajo tiene como **objetivo general** desarrollar un Sistema de Información Geográfica Web que permita mejorar la gestión de la información de los datos espaciales en el Centro Meteorológico Provincial.
Para ello se debe dar cumplimiento a los siguientes **objetivos específicos**:

- Analizar el proceso de gestión de la información geográfica en el CMP.
- Diseñar un sistema que permita la gestión de la información geográfica en el CMP.
- Implementar el sistema propuesto mediante una aplicación Web que se ajuste a las particularidades del centro.
- Validar el sistema informático.

Como **tareas a realizar** se plantean las siguientes:

- Entrevistas a los especialistas del grupo de pronóstico y climatológico.
- Investigación de estudios sobre los datos espaciales y servidores de mapas.

- Selección de las metodologías, lenguajes, sistema gestor de base de datos y herramientas a utilizar.
- Definición de los requisitos funcionales y no funcionales.
- Modelación de la base de datos del sistema.
- Confección de la documentación de la información generada durante el análisis, diseño e implementación del sistema.

La información espacial a la que se quiere acceder debe estar acorde con ciertas normas y estándares y los recursos informáticos con especificaciones, protocolos e interfaces que garanticen la interoperabilidad. Como los datos cambian con rapidez al igual que la forma en que se almacenan es necesario que los sistemas sean interoperables y la información pueda ser accedida por diferentes aplicaciones.

Esta investigación está dedicada a la concepción del sistema y para mejor comprensión está estructurada en tres capítulos, además de las Conclusiones, Recomendaciones, Referencias Bibliográficas, Bibliografías, Glosario de términos y Anexos.

Capítulo 1 "Fundamentación Teórica": En este capítulo se realiza un estudio acerca de los sistemas de información geográficos en general y los aplicados al campo de acción. Se analiza el proceso de gestión de la información geográfica en el Centro Meteorológico Provincial. Se muestra una descripción de las tecnologías, lenguajes, herramientas actuales sobre las que se apoyará la idea a defender.

Capítulo 2: Planificación y control para el desarrollo del software. En este capítulo se hace una descripción del sistema a automatizar, en el cual se mostrará el diagrama de casos de uso del sistema para su mejor entendimiento. Se definirán los actores del sistema, así como los requerimientos funcionales y no funcionales. Además se describen los patrones de diseño y de arquitectura que se tuvieron en cuenta durante el desarrollo de la aplicación y se evidencia su utilización. Se realiza el diagrama de clases del diseño para cada caso de uso, así como los diagramas de secuencia por escenario de los mismos.

Capítulo 3: Construcción y Validación: En este capítulo se presenta el diagrama de componentes. Se muestran los estándares de programación para el código fuente de la aplicación y se realiza el análisis de factibilidad y la validación del producto.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

Introducción

En este capítulo se realiza un estudio acerca de las características de los sistemas de información geográficos en general. Se analizaron algunos de los aplicados al campo de la meteorología y climatología. Se analiza el proceso de gestión actual de la información geográfica en el Centro Meteorológico Provincial. Se muestra una descripción de las tecnologías, lenguajes y herramientas actuales.

1.1 Características de los sistemas de Información geográficos (SIG).

El criterio de diferentes autores definen a los sistemas de información geográficos como:

"...una tecnología computacional compuesta por equipo (hardware), programas (software) y datos, empleados para capturar, editar, y lo más importante analizar información geográfica" [3].

"paquetes de programación que alternan el almacenamiento, administración, análisis, modelaje y mapeo de estadística en formato digital"[3].

"colección organizada de equipo de cómputo, paquetería, datos geográficos y personal calificado para capturar, almacenar, manipular, analizar y desplegar eficientemente toda la información geográficamente referenciada"[3].

Sin embargo creemos que, tal y como sostiene Burrough, un SIG debe verse también como un modelo del mundo real, por lo que se podría definir como:

"Modelo informatizado del mundo real, en un sistema de referencia ligado a la Tierra para satisfacer unas necesidades de información concretas".

En un sentido más genérico, los SIG son herramientas que permiten a los usuarios crear consultas interactivas, analizar la información espacial, editar datos, mapas y presentar los resultados de todas estas operaciones.

1.1.1 Tipos de datos de un SIG

Aunque existen diferentes representaciones y formatos de los tipos de datos en los SIG estos pueden clasificarse en dos grandes grupos:

- Datos espaciales
- Datos de atributos

Los datos espaciales, también llamados geográficos o de ubicación, son los objetos geográficos de un mapa, tales como límites políticos, ríos, carreteras. Se encuentran generalmente en formato de vector y raster. [4]

Los SIG que se centran en el manejo de datos en formato vectorial son más populares en el mercado. No obstante, los SIG raster son muy utilizados en estudios que requieren la generación de capas continuas, necesarias en fenómenos no discretos y en estudios medioambientales donde no se necesite una excesiva precisión espacial (contaminación atmosférica, distribución de temperaturas, localización de especies marinas y análisis geológicos). [5]

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. Por ejemplo, una base de datos que describe los lagos, puede contener datos sobre la batimetría. [5]

Los datos de atributo son variables que representan características de datos espaciales y entidades geográficas[4]. Podrían ser el valor de una variable vista en una estación meteorológica, la altura de un área geográfica o la cantidad de personas que viven en un municipio. Se basa en el Modelo Relacional de Bases de Datos.

1.1.2 Funcionamiento de los SIG

La razón fundamental para utilizar un SIG es la gestión de información espacial. El sistema permite separar la información en diferentes capas temáticas y las almacena independientemente, permitiendo trabajar con ellas de manera rápida y sencilla, facilitando al profesional la posibilidad de relacionar la información existente. En estos el orden de superposición de la información es muy importante para obtener visualización correcta del mapa final. [5]

1.1.3 Estándares WEB que garantizan la interoperabilidad.

La consolidación del OpenGis Consortium's (OGC) como el ente encargado de la estandarización del manejo de la información geográfica WEB, ha permitido el desarrollo de herramientas que permiten la visualización de información geoespacial proveniente de varias organizaciones y servidores a través de la World Wide Web, para llegar a realizar el sueño de

verdadera interoperabilidad y difundir una especificación de cartografía en Web que sea adoptada y promulgada por los vendedores. [6]

Existen entidades reguladoras que estandarizan el acceso a información geográfica en la web:

- ISO (Organización Internacional de Estándares) produce estándares y especificaciones generales para guiar la implementación.
- Consortium (W3C) World Wide Web
- OpenGIS Consortium (OGC) conformado por socios industriales, gubernamentales y universitarios. La principal misión del consorcio ha sido intentar estandarizar los servicios relacionados con la geografía en entornos distribuidos. [6] Es una organización dedicada a la definición de estándares abiertos e interoperables.
- Normas de organizaciones nacionales

La arquitectura que están utilizando en el OGC define un número elevado de servicios, algunos de los cuales están sin especificar, todo ello con el objetivo final de permitir el ensamblado de los servicios en entornos distribuidos. Para poder satisfacer este requisito hace falta que los servicios sean interoperables. Las especificaciones del OGC más consolidadas y validadas son: WMS y WFS cuyo formato nativo de datos es GML. Algunas de estas especificaciones datan de los años 90 y están definidas de forma que el servicio se ofrezca a través del protocolo HTTP con una notación de parámetros orientada a un script o cgi. Todos los servicios poseen un método por el que se describe las capacidades del mismo en forma de fichero marcado en XML. [7]

Los servicios de SIG en línea se acceden a través de los catálogos en línea que conducen a datos espaciales por medio de metadatos.[6]

1.1.4 Servicios WMS y WFS

El servicio de mapas en la Web o WMS de OGC permite la visualización de una imagen cartográfica generada a partir de una o varias fuentes: mapa digital, datos de un SIG u orto foto, provenientes de uno o varios servidores, ficheros de datos o bases de datos geográficos.

El servicio de fenómenos en la Web o WFS de OGC permite acceder a los datos mismos, mediante el empleo del formato GML de OGC. Así puede acceder al archivo que define la geometría de un objeto cartográfico, como un río, una ciudad, una parcela y disponer de esa información vectorial en el propio ordenador.

1.2 Sistemas de información geográficos aplicados a la meteorología y la climatología.

Los SIG tienen muchas aplicaciones que están surgiendo en la actualidad. En el área de la climatología algunos están creados para gestionar información relacionada con evolución de frentes, anticiclones, borrascas, simulaciones y evolución para predicciones atendiendo a la altura, velocidad del viento, presión, temperatura entre otras.

En el área de la meteorología los SIG están creados para gestionar datos de observaciones meteorológicas, conocer la trayectoria de los organismos pasados y actuales

En la actualidad se difunden a través de la WEB capas de información con salida en formato de SIG. Por mencionar algunos y su distribución de datos tenemos:

- Observaciones en tiempo real y productos de predicción de la NOAA.
- Sistema de información geográfico de trayectoria de organismos.
- Datos históricos de organismos en la cuenca del pacífico y del atlántico.
- Datos en SIG de trayectoria de organismos tiempo real del Centro Nacional de Huracanes.
- Satélite geoestacionario en sistema de información geográfico.

Aunque estos recursos son abundantes en la WEB no todos presentan la totalidad de los datos. Por ejemplo el sistema de información geográfico de trayectoria de organismos solo muestra datos en el período de 1951 a 2012, faltando las temporadas actuales. Un inconveniente de estos sistemas es que se encuentran en servidores externos al CMP y no se tiene seguridad de poder acceder siempre a ellos en todo momento debido a que pueden quedar inutilizados sin previo aviso por parte de los administradores. Otro problema es que las observaciones en tiempo real son hechas en posiciones externas a Cuba y no consideran los datos de la red de estaciones meteorológicas de Cuba. Podemos afirmar entonces que los sistemas actuales no están adecuados a las condiciones actuales.

1.3 Proceso de gestión de la información geográfica en el CMP de Cienfuegos.

En el Centro Meteorológico Provincial de Cienfuegos se trabaja a diario con información geográfica. En el departamento climatológico por ejemplo se realizan las guías climáticas de la provincia. Para realizar las guías los especialistas utilizan información que se genera en diferentes sitios geográficos, como por ejemplo las estaciones meteorológicas, datos de

pluviómetros emitidos por recursos hidráulicos, datos de estaciones automáticas entre otras. Se utiliza generalmente el SIG de escritorio MapInfo para realizar los mapas temáticos como pueden ser el de la sequía o la temperatura.

Por su parte el grupo de pronóstico necesita conocer la trayectoria pasada de algún organismo en la cuenca del atlántico, su probable trayectoria e información de la velocidad de sus vientos, presión, categoría entre otros. En ocasiones necesitan conocer la trayectoria de organismos de otras temporadas. Para realizar esta labor los especialistas consultan imágenes a través de la WWW y esta información a medida que es descargada se almacena en un portal. Como la información puede provenir de distintas fuentes se almacena en diferentes formatos de imágenes, se guardan entonces grandes cantidades de ficheros de imágenes. A simple vista se puede inferir la posición de algún organismo a través de estas imágenes pero no es posible determinarla de forma exacta. Si tratas de determinar la posición el margen de error sería muy elevado. Esto se debe a que se almacenan las imágenes que no están georeferenciadas en vez de almacenar el dato exacto de la trayectoria del organismo.

Los datos en ocasiones se encuentran en diferentes máquinas del CMP, esto trae como consecuencia que la información geográfica esté repetida o exista redundancia en sus datos.

Aunque en el CMP a diario se trabaja con información geográfica, ésta se encuentra distribuida de forma no ordenada, en diferentes formatos y son escasas las herramientas para visualizar y analizar la información espacial. El proceso para gestionar la información geográfica se hace tedioso y no es operativo.

1.4 Metodologías para desarrollo de software

Desarrollar un software puede ser un reto , pues su creación en el menor tiempo posible y con calidad es casi imposible si no se cuenta con algún proceso que ayude a agilizar el desarrollo del mismo.

Desde hace algún tiempo se vienen utilizando las metodologías, ya que imponen un proceso disciplinado en el desarrollo, con el fin de hacerlo más predecible y eficiente. En este proceso se define quien está haciendo qué, cuándo y dónde para alcanzar un determinado objetivo. En la ingeniería el objetivo es construir un producto de software o mejorar uno existente con calidad. [8]

Dentro de las metodologías de desarrollo de software se encuentran algunas como las que se caracterizan a continuación:

RUP- Rational Unified Process

Es un proceso para el desarrollo de software que define claramente quien, cómo, cuándo y qué debe hacerse en el proyecto. Presenta 3 características esenciales está dirigido por los casos de uso que orientan el proyecto a la importancia para el usuario y lo que este quiere, está centrado en la arquitectura que relaciona la toma de decisiones que indican cómo tiene que ser construido el sistema y en qué orden, es iterativo e incremental, divide el proyecto en mini-proyectos donde los casos de uso y la arquitectura cumplen sus objetivos de manera más depurada. [8]

Extreme Programming (XP)

La programación extrema centra su atención en la producción de software con una fuerte arquitectura, intentando sacar productos al mercado rápidamente, con gran calidad y motivando al equipo de trabajo para seguir mejorando esta tendencia.

Como metodología, la programación extrema, presenta muchos puntos comunes con el desarrollo incremental, comenzando por el hecho de que el software desarrollado con XP se realiza de forma incremental. Cuenta con cuatro fases fundamentales: codificar, probar, escuchar y diseñar. [8]

OpenUp

Dentro de las metodologías existentes se ha decidido utilizar esta metodología debido a que permite mantener la filosofía de RUP que es una de las más empleadas actualmente en la universidad. Desarrollada por IBM y reconocida mundialmente como uno de los procesos de desarrollo de software de mayor calidad, basándose en los principios de adaptación, importancia a los involucrados e interesados en los resultados del proyecto, colaboración, valor a la iteración y calidad continua. Permite un abordaje ágil al proceso de desarrollo de software con sólo proveer un conjunto simplificado de contenidos, fundamentalmente relacionados con orientación, productos de trabajo, roles y tareas. Es un proceso interactivo de desarrollo de software simplificado, completo y extensible.[9]

Es un framework de procesos de desarrollo de software de código abierto, es un proceso modelo y extensible, dirigido a gestión y desarrollo de proyectos de software basados en desarrollo iterativo, ágil e incremental y es aplicable a un conjunto amplio de plataformas y aplicaciones de desarrollo.

Se centra en particular la arquitectura para facilitar la colaboración técnica, reducir el riesgo y minimizar el esfuerzo de desarrollo. Procura un equilibrio entre las necesidades de los involucrados con los resultados del proyecto y los costos técnicos. Desarrolla un ciclo de vida interactivo que mitiga el riesgo a tiempo y ofrece demostrar resultados en curso al cliente del proyecto.[9]

Beneficios en el uso del OpenUp

Es apropiado para proyectos pequeños y de bajos recursos permite disminuir las probabilidades de fracaso en estos e incrementar las probabilidades de éxito.

Permite detectar errores tempranos a través de un ciclo iterativo.

Evita la elaboración de documentación, diagramas e iteraciones innecesarios requeridos en la metodología RUP.

Por ser una metodología ágil tiene un enfoque centrado al cliente y con iteraciones cortas.

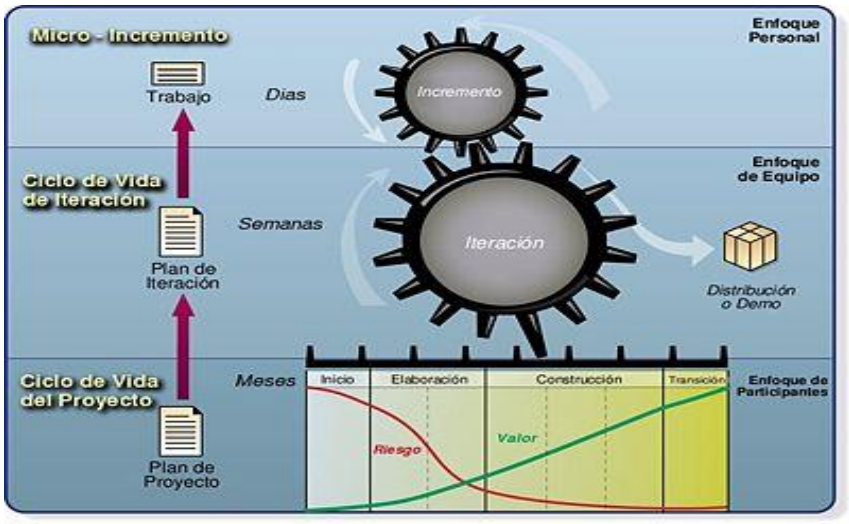


Figura 1. Ciclo de vida de Open Up

Fases de OpenUP

1. Concepción

Primera de las cuatro fases del ciclo de vida del proyecto, se basa en el entendimiento del propósito y objetivos. Recopila suficiente información para confirmar lo que el proyecto debe hacer. [10]

2. Elaboración

Es la segunda de las cuatro fases del ciclo de vida, donde se tratan los riesgos significativos para la arquitectura. Su propósito es establecer la base para la elaboración de la arquitectura del sistema.[10]

3. Construcción

Esta fase está enfocada al diseño, implementación y prueba de las funcionalidades para desarrollar un sistema completo. El propósito de esta es completar el desarrollo del sistema basado en la arquitectura definida. [10]

4. Transición

Es la última fase, cuyo propósito es asegurar que el sistema es entregado a los usuarios, evalúa la funcionalidad y performance del último entregable de la fase de construcción.[11]

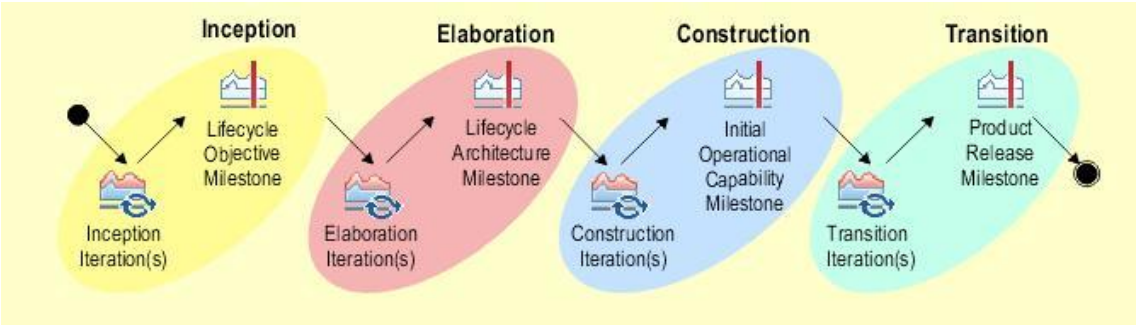


Figura 2. Iteraciones de Open Up

Roles en la solución

- **Analista**

La persona en este cargo representa al cliente y las preocupaciones de los usuarios finales mediante la recopilación de los grupos interesados para entender el problema a resolver y por la captura y fijación de prioridades para los requisitos.[12]

- **Desarrollador**

Es responsable del desarrollo de una parte del sistema, incluyendo el diseño que se ajuste a la arquitectura, posible creación de un prototipo de la interfaz de usuario, la aplicación, la unidad de pruebas y la integración de los componentes que forman parte de la solución. [12]

1.5 Arquitectura de Software

La arquitectura de un sistema de programa o computación es la estructura de las estructuras del sistema, la cual comprende los componentes del software, las propiedades de esos componentes visibles externamente, y las relaciones entre ellos. [13]

La arquitectura destaca decisiones tempranas del diseño que tendrían un profundo impacto en todo el trabajo de ingeniería del software que sigue. Facilita la comunicación entre todas las partes (partícipes) interesadas en el desarrollo del sistema. Constituye un modelo relativamente pequeño e intelectual comprensible de cómo está estructurado el sistema y de cómo trabajan juntos sus componentes. [13]

Arquitectura basada en componentes

Los sistemas de software basados en componentes se basan en principios de una ingeniería de software específica (ISBC). La ingeniería de software basada en componentes ha cambiado la forma en que se desarrollan los sistemas de software. El programador cambia sus objetivos y pasa de programar el software a componer sistemas de software, para su posterior utilización como componentes de software reutilizables.

El objetivo de esta arquitectura es construir aplicaciones complejas mediante ensamblado de componentes que han sido previamente diseñados por otros desarrolladores o por subgrupos del equipo de desarrollo, a fin de ser rehusados en múltiples aplicaciones. En la arquitectura

basada en componentes la interfaz constituye el elemento básico de conexión. Cada componente debe describir de forma completa las interfaces que ofrece, así como las interfaces que requiere para su operación. Además, debe operar con independencia de los mecanismos internos que utilice para soportar la funcionalidad de la interfaz. [14]

Patrones de arquitectura, casos de uso y diseño.

¿Qué es un patrón?

Según: **Christopher Alexander**.[1]

"*Cada patrón describe un problema que ocurre una y otra vez en nuestro entorno, para describir después el núcleo de la solución a ese problema, de tal manera que esa solución pueda ser usada más de un millón de veces sin hacerlo siquiera dos veces de la misma forma*".

Patrones de arquitectura

Los patrones de arquitectura describen un problema particular y recurrente de diseño, que aparece en contextos de diseño específico, y presenta un esquema genérico demostrado con éxito para su solución. Algunos patrones de arquitectura son:

1. Capas: es utilizado para dividir sistemas de software complicados, donde la capa más alta, es decir la superior, necesita de los servicios de la capa baja para su realización, pero la última capa es ajena a la capa superior.

2. Tubería-filtros: enfatiza la transformación incremental de los datos a través de los sucesivos componentes.

3. Pizarra: es utilizado en sistemas de conocimientos como lo son sistemas expertos y sistemas multiagentes.

4. Bróker: es un patrón muy útil en la estructuración de sistemas distribuidos con componentes desacoplados que interactúan por medio de invocación a servicios remotos.

---

[1]**Christopher Alexander** (nacido el 4 de octubre de 1936 en Viena, Austria) es un arquitecto, reconocido por sus diseños destacados de edificios en California, Japón y México

5. Presentación-Abstracción-Control: este patrón se utiliza como una estructura jerárquica de los agentes, cada uno de ellos consiste en una tríada de presentación, la abstracción y el control de partes.

6. Reflexión: este patrón proporciona un mecanismo para cambiar la estructura y comportamiento del software dinámicamente.

7. Microkernel: este patrón es aplicable a sistemas de software que deben ser habilitados para adaptarse a requerimientos cambiantes del sistema.

8. Modelo-Vista-Controlador

De acuerdo con la definición y la función que realizan estos patrones se ha decido utilizar en el diseño del módulo el siguiente: Modelo-Vista-Controlador (MVC).

En ocasiones se le define más bien como un patrón de diseño o como práctica recurrente, y en estos términos es referido en el marco de la estrategia arquitectónica de Microsoft. El patrón MVC separa el modelado del dominio, la presentación y las acciones basadas en datos ingresados por el usuario en tres clases diferentes:

- Modelo: El modelo administra el comportamiento y los datos del dominio de aplicación, responde a requerimientos de información sobre su estado (usualmente formulados desde la vista) y responde a instrucciones de cambiar el estado (habitualmente desde el controlador).

- Vista: Maneja la visualización de la información.

- Controlador: Interpreta las acciones del ratón y el teclado, informando al modelo y/o a la vista para que cambien según resulte apropiado.

Patrones de Caso de uso.
Los patrones de casos de uso ayudan a identificar casos de uso a partir de las características del negocio o de los requerimientos del sistema de una forma más eficiente. Entre los patrones de casos de uso, se encuentran:

CRUD (Create, Read, Update, Delete) o Gestión de Información: Modela las operaciones que pueden ser realizadas sobre una parte de la información de un tipo específico, tales como creación, lectura, actualización y eliminación.

Concordancia Reuso: Modela la reutilización del modelo que consiste en tres casos de uso. La primera sub-secuencia, llamada Común, modela la secuencia de las acciones que debe aparecer en múltiples casos de empleo en el modelo. Otros casos de empleo modelan los usos del sistema que comparten la sub-secuencia común de acciones. Obviamente, habrá al menos dos de ellos.

Inclusión o extensión concreta: El modelo de extensión consiste en dos casos de empleo y una relación ampliada entre ellos. El caso de empleo de extensión es concreto, es decir que puede estar instanciado, solo así se amplía el caso de empleo bajo. Este último puede ser hormigón o resumen.

Patrones de Diseño

Un patrón de diseño es una descripción de clases y objetos comunicándose entre sí adaptada para resolver un problema de diseño general en un contexto particular. Además de permitir describir el sistema con el suficiente detalle para ser implementado.

Los patrones de diseño tienen como ventaja que proponen una forma de reutilizar la experiencia de los desarrolladores, para ello clasifican y describen formas de solucionar problemas que ocurren de forma frecuente en el desarrollo. Además están basados en la recopilación del conocimiento de los expertos en desarrollo de software por lo que es una experiencia real, probada, que funciona y ayuda a no cometer los mismos errores.

Entre los patrones más empleados se encuentra **GRASP** (las siglas pertenecen a las palabras del inglés, General Responsibility Assignment Software Patterns) en el que se destacan 5 patrones fundamentales:

- **Experto:** Es un patrón que se usa más que cualquier otro al asignar responsabilidades, es un principio básico que suele ser útil en el diseño orientado a objetos. El cumplimiento de una responsabilidad requiere a menudo información distribuida en varias clases de objetos. El

patrón Experto asigna responsabilidades a las clases que tienen la información necesaria para cumplir con la responsabilidad. [15]

- **Creador:** Guía la asignación de responsabilidades relacionadas con la creación de objetos, tarea muy frecuente en los sistemas orientados a objetos. El propósito fundamental de este patrón es encontrar un creador que debemos conectar con el objeto producido en cualquier evento. Al escogerlo como creador, se da soporte al bajo acoplamiento. Lo que define este patrón es que una instancia de un objeto la tiene que crear el objeto que tiene la información para ello. [15]

- **Alta Cohesión:** Mantiene la complejidad dentro de los límites manejables, es decir asigna una responsabilidad de modo que la cohesión siga siendo alta. La cohesión es una medida de cuan relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme. [15]

- **Controlador:** es un patrón que sirve como intermediario entre una determinada interfaz y el algoritmo que la implementa, de tal forma que es la que recibe los datos del usuario y la que los envía a las distintas clases según el método llamado. Asigna las responsabilidades de capturar los eventos del sistema a las clases. [15]

Dentro de los patrones de diseño podemos encontrar patrones **Gof (gang of four o pandilla de los cuatros)**dentro de los cuales se encuentran:

- **Patrones de creación:**

  - **Constructor:** separa la construcción de un objeto complejo de su representación, de forma que el mismo proceso de construcción pueda crear diferentes representaciones.
  - **Método de la fábrica:** este patrón define una interfaz para crear un objeto, pero deja que sean las subclases quienes decidan qué clase instanciar. Además permite que una clase delegue en sus subclases la creación de objetos.
  - **Prototipo**: especifica los tipos de objetos a crear por medio de una instancia de un prototipo, y crea nuevos objetos copiando este mismo prototipo.

- **Patrones estructurales**

  - **Adaptador:** convierte la interfaz de una clase en otra distinta que es la que esperan los clientes. Además permiten que cooperen clases que de otra manera no podrían hacerlo, pues tendrían interfaces incompatibles
  - **Puente**: desvincula una abstracción de su implementación, de manera que ambas puedan variar de forma independiente.
  - **Compuesto:** combina objetos en estructuras de árbol para representar jerarquías de parte-todo. Permite que los clientes traten de manera uniforme a los objetos individuales y a los compuestos.
  - **Decorador:** añade dinámicamente nuevas responsabilidades a un objeto, proporcionando una alternativa flexible a la herencia para extender funcionalidades.
  - **Fachada:** proporciona una interfaz unificada para un conjunto de interfaces de un subsistema. Define la interfaz de alto nivel que hace que sea más fácil de utilizar el subsistema.

- **Patrones de comportamiento**

  - **Cadena de Responsabilidad:** evita acoplar el emisor de una petición a su receptor, al dar más de un objeto la posibilidad de responder a la petición. Crea una cadena con los objetos receptores y pasa la petición a través de la cadena hasta que esta sea tratada para un objeto.
  - **Iterador:** proporciona un modo de acceder secuencialmente a los elementos de un objeto agregado sin exponer su representación interna.
  - **Mediador:** define un objeto que encapsula cómo interactúan un conjunto de objetos. Promueve un bajo acoplamiento al evitar que los objetos se refieran unos a otros explícitamente y permite variar la interacción entre ellos de forma independiente.
  - **Observador:** define una dependencia de uno a muchos entre objetos, de forma que cuando un objeto cambia de estado se notifica y actualizan todos los objetos automáticamente.
  - **Estado:** permite que un objeto modifique su comportamiento cada vez que cambia su estado interno. Tal parece que cambia la clase del objeto. [15]

1.6 Tecnologías y herramientas a emplear

1.6.1 Sistema Gestor de Bases de Datos Geográficos PostgreSQL/PostGIS.

PostGIS es una extensión al sistema de base de datos objeto-relacional PostgreSQL, que permite el uso de objetos geoespaciales como: líneas, polígonos, multilíneas, multipuntos y colecciones geométricas, convirtiéndola en una base de datos espacial para su utilización en Sistemas de Información Geográfica. Ha demostrado ser superior a la extensión geográfica de la nueva versión de MySQL. Un aspecto que se tiene en cuenta a la hora de utilizar PostGIS es su interoperabilidad con otros sistemas certificados en el 2006 por el OGC. Actualmente hay diferentes productos informáticos que utilizan PostGIS como apoyo a su base de datos el OpenGIS, Quantum GIS, GeoServer, Mapserver, OpenJump, GvSIG. Se publica bajo la licencia pública general de GNU, que tiene como propósito declararlo como software libre.

1.6.2 MapServer servidor de mapas en la WEB.

Es un entorno de desarrollo de código abierto para la creación de aplicaciones SIG con el fin de visualizar, consultar y analizar información geográfica a través de la red mediante la tecnología web. Este puede ser utilizado como una aplicación CGI (del inglés, Common Gateway Interface) o a través del acceso a la API (del inglés, Application Programming Interface) de MapServer que proveen las librerías Mapscript. Entre sus características podemos mencionar, que entre los tipos de información geográfica soporta formatos vectoriales tales como: ESRI shapefiles, PostGIS, ESRI ArcSDE, GML y otros vía OGR; además de formatos raster como: JPG, PNG, GIF, TIFF/GeoTIFF, EPPL7 y otros vía GDAL. La configuración de los mapas se hace a través de los ficheros MapFile, estos pueden ser estáticos o dinámicos.

De acuerdo a los tipos de licencias de los componentes y herramientas que se proponen a utilizar se puede catalogar legalmente esta arquitectura de modelo libre, permitiendo la utilización, modificación y distribución de las mismas por parte del equipo de desarrollo, sin necesidad de obtener la autorización de sus respectivos titulares.

1.6.3 Open Layer

OpenLayers es una biblioteca de JavaScript de código abierto, para la elaboración de mapas web interactivas que se puede leer en casi cualquier navegador. Dado que es una biblioteca

del lado del cliente, no requiere software del lado del servidor especial o ajustes que puedan utilizarlo sin siquiera descargar nada. Originalmente desarrollado por Metacarta, como respuesta, en parte, a Google Maps, se ha convertido en un marco maduro, popular con muchos desarrolladores apasionados y una comunidad muy útil. [16]

1.6.3.1 JavaScript

JavaScript se implementó por primera vez en la versión beta de Netscape Navigator 2.0 en junio de 1995, por la empresa Netscape Communications Corporation. Es un lenguaje de programación interpretado utilizado en la realización de páginas web. Una de sus principales características es que es orientado a objetos. La verdadera fuerza de JavaScript es la compatibilidad con los distintos navegadores, incluso con los más anticuados. Además de que es interpretado por todos los navegadores modernos. [16]

1.6.4 UML 2.0

El Lenguaje Unificado de Modelado (UML) es un lenguaje para visualizar, especificar, construir y documentar los artefactos de un sistema. Es la especificación de OMG (Grupo de gestión de objetos, en inglés Object Management Group) y no solo es la forma mundial de representar la estructura de las aplicaciones, comportamiento y arquitectura sino que también representa procesos de negocio y estructura de datos. Permite el modelado de sistemas con tecnología orientada a objetos y no es considerado un proceso.

Este lenguaje tiene elementos que conforman su vocabulario. En los diagramas se relacionan estos conjuntos de elementos y son los encargados de visualizar un sistema desde diferentes perspectivas. [16]

1.6.5 Herramienta CASE

Son muy utilizadas en la actualidad porque son aplicaciones que ayudan a aumentar la productividad del desarrollo de software reduciendo costes, tiempo y dinero de los sistemas en confección. El uso de las herramientas CASE presenta para el usuario que las utiliza varias ventajas entre las que se encuentra que ayuda a mejorar la calidad del software.

En el mundo hoy existen varias herramientas CASE, entre ellas se encuentra el Visual Paradigm, herramienta que será utilizada para la confección del producto.

1.6.5.1 Visual Paradigm for UML 8.0

Es una herramienta CASE, desarrollada por la compañía Visual Paradigm International, que utiliza UML como lenguaje de modelado. Presenta las siguientes características:

- Soporta el ciclo de vida completo del software: análisis, diseño, implementación y despliegue.
- Permite la captura de requisitos, el dibujo de diagramas UML, la realización de ingeniería inversa y generación de código PHP.
- Se integra con las siguientes herramientas:
  - Eclipse/IBM WebSphere
  - Builder
  - Net Beans IDE
  - Oracle JDeveloper
  - BEA Weblogic
- Está disponible en varias ediciones, cada una destinada a necesidades específicas: Enterprise, Professional, Community, Standard, Modeler y Personal.
- Es soportado por varios Sistemas operativos entre los que se encuentra Ubuntu, Windows y Mac OS.

1.6.6 Marco de desarrollo para la aplicación web

Los marcos de trabajo o framework de desarrollo web son muy utilizados debido a que representan una estructura de software compuesta de componentes personalizables e intercambiables para el desarrollo de una aplicación que ayuda a la rápida confección de sitios web. Los frameworks desde su creación han sido muy utilizados por los programadores y diseñadores porque los provee de una mejor estructura y organización en sus proyectos. Para las aplicaciones desarrolladas en PHP existe una cantidad considerable de frameworks libres de mucha calidad y renombre que pueden ser utilizados entre los que

se pueden encontrar a Zend Framework, y Symfony entre muchos otros. En esta investigación se hablará del marco de desarrollo ExtJS, debido a su uso en la implementación de los componentes referidos en el documento.

1.6.6.1 ExtJS 3.2

ExtJS es una librería de Java Script para el desarrollo de aplicaciones enriquecidas para la web haciendo un uso extensivo de las tecnología AJAX, además de XHTML/ DHTML y DOM. Su potencia radica en colección de componentes para el diseño de interfaces de usuario (GUIs). En principio Extjs fue creado como una extensión de Yahoo User Interface (YUI) otro framework bastante similar, Extjs incluye interoperabilidad con jQuery, Prototype y Scrpt.aculo.US. Entre sus principales características se encuentran:

- 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 intuitivo y fácil de utilizar.
- Distribuido bajo licencias Open Source y comerciales.

1.6.7 NetBeans IDE 7.1.

El NetBeans es un entorno de desarrollo integrado (IDE), por sus siglas en inglés (Itegrated Development Environment) es una herramienta para programadores pensada para escribir, compilar, depurar y ejecutar programas. Está escrito en Java, pero puede servir para cualquier otro lenguaje de programación. Existe además un número importante de módulos para extender el IDE NetBeans. El IDE NetBeans es un producto libre y gratuito sin restricciones de uso. (8)Tiene buen soporte de Portal en Perl para Automatización de Web (Webapps) (war, jsp y Servlets) y además en la actualidad cuenta con el plugin *Portal Pack* para la realización de Portlets.

Características:

- Generación automática de código.
- Utilidades de búsqueda y reemplazo muy intuitivos, amigables y sobre todo realmente útiles para el programador.
- Coloración de código.

- Refactorización inteligente.
- Diseñadores de interfaces gráficas para Swing y JSF.
- Asistentes para escritura de código rápido de acceso a datos basados en JPA (Java Persistence API).
- Asistentes para escritura de servicios Web.
- Soporte para múltiples máquinas virtuales.
- Soporte para múltiples servidores contenedores J2EE.
- Soporte para librerías de terceros.
- También es importante recordar la variedad de plugins (componentes de Software conectables a NetBeans) disponibles para incrementar las funcionalidades del IDE.

Conclusiones del Capítulo

Para dar solución a la problemática planteada se decidió hacer uso de la metodología de desarrollo OpenUP, el lenguaje de modelado visual UML 2.0, la herramienta CASE Visual Paradigm 8.0 para la confección de los diagramas y como lenguaje de programación, JavaScript. Como framework de desarrollo ExtJS y para la implementación el uso del IDE NetBeans.

CAPÍTULO 2: PLANIFICACIÓN Y CONTROL PARA EL DESARROLLO DEL SOFTWARE

Introducción

Entre los datos y fuentes de datos puestos a disposición al Centro Meteorológico Provincial se encuentran:

- Datos de trayectoria de tormentas y huracanes del atlántico en el periodo de 1851 al 2013. Este incluye los puntos de cada organismo la línea de trayectoria y el radio de los vientos.
- Datos geográficos de origen global imágenes satelitales, costas y límites de países.
- Datos Geográficos de Cienfuegos municipios, núcleos urbanos, mapas de elevación, hidrografía entre otros.
- Fuentes de Datos del Centro Nacional de Huracanes en Formato Sistema de Información Geográfico (NOAA)
- Archivo de datos del NHC (NOAA)
- Información de tiempo real para las imágenes satelitales e infrarrojo.

Arquitectura particular del IDE configurado en el Centro Meteorológico Provincial de Cienfuegos:

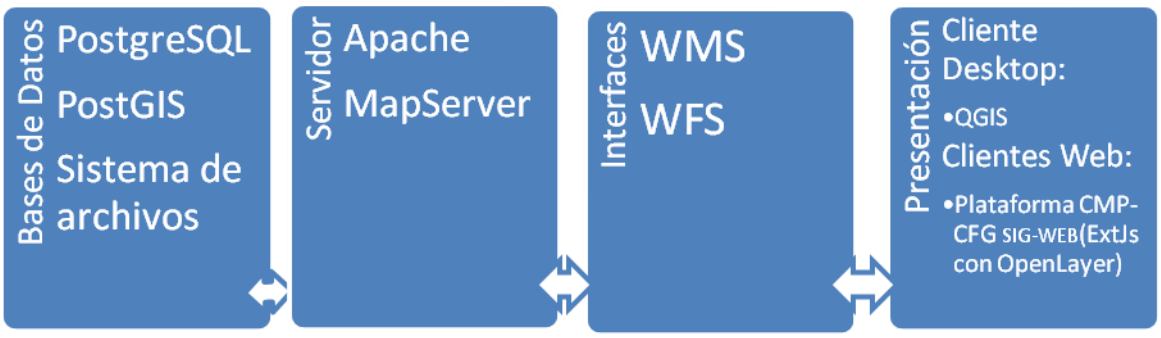


Figura 3

Cliente web

El Sistema de Información Geográfica del Centro Meteorológico Provincial de Cienfuegos es un cliente para obtener los servicios WMS y WFS pero a través de la web. Está desarrollado usando ExtJs y OpenLayer. Aunque los usuarios del centro meteorológico pueden acceder a

los servicios con un simple navegador web con dicha herramienta pueden visualizar las capas de información de forma organizada y con un ambiente amigable.

2.1. Normalización de archivo hurdat2

Se normalizó la base de datos de tormentas y huracanes en el período de 1851 a 2013. Dichas relaciones se encuentran en PostgreSQL/PostGIS. Se almacenan los puntos en latitud y longitud de cada organismo y su información en cada registro de tiempo. Se crearon 3 tablas para almacenar los puntos que representan la posición del organismo, otra tabla con las líneas que representa la trayectoria y por último una tabla de polígonos que almacena el radios de los vientos. El fichero de datos normalizado es hurdat2-atlantic-1851-2013 (hurdat2) y presenta la siguiente estructura:

Está formado por dos tipos de líneas, las líneas de cabecera y las líneas de datos. Todas ellas delimitadas por comas. Las líneas de cabecera tienen el siguiente formato:

Los primeros dos caracteres corresponden a la región del organismo, estos valores pueden ser *AL* Atlántico, *EP* Este del pacífico y *CP* Pacifico Central. Los dos siguientes caracteres muestran el número del organismo en el año. Los cuatro siguientes caracteres representan el año. Luego el delimitador coma y a continuación el nombre del organismo. Después del nombre del organismo el delimitador coma y la cantidad de puntos de datos que presenta dicho organismo en ese año. Ejemplo de la línea de cabecera para el huracán Sandy:

AL182012, SANDY, 45,

A continuación para cada línea de cabecera corresponden la línea de datos y su cantidad es determinada por la cantidad de puntos que presentó dicho organismo. Para el ejemplo del huracán Sandy corresponden 45 líneas de datos las cuales presentan el siguiente formato:

20121025, 0600, HU, 20.1N, 76.0W, 100, 954, 240, 240, 70, 120, 50, 60, 40, 40, 25, 20, 20, 20,

Los primeros cuatro caracteres corresponde al año, luego siguen dos caracteres que representa el mes, y los dos siguientes el día que se tomó dicho dato. Luego el delimitador coma con un valor de cuatro cifras que corresponde a la hora UTC. A continuación la clasificación del organismo que puede ser *TD* Depresión Tropical, *TS* Tormenta Tropical, *HU* Huracán, *EX* Ciclón Extra tropical, *SD* Depresión Sub tropical, *SS* Tormenta Sub Tropical, *LO* Baja o *DB* Disturbio. Luego se presenta la latitud y longitud separada por coma. Los

vientos máximos sostenidos y la presión en milibar le continúan. Los valores de presión, los vientos en cada uno de los cuatro cuadrantes. Los cuatro primeros valores corresponden al radio de vientos de 34 nudos que se extienden en el cuadrante *NE*, *SE, SW, NW* expresados en millas náuticas. Los cuatro siguientes valores corresponden al radio de vientos de 50 nudos que se extienden en el cuadrante *NE, SE, SW, NW* expresados en millas náuticas. Los cuatro últimos valores corresponden al radio de vientos de 64 nudos que se extienden en el cuadrante *NE, SE, SW, NW* expresados en millas náuticas. Para más información ver formato de archivo (hurdat2).

El servidor de mapas MapServer la cuál brinda servicios WFS y WMS permite poder brindar los datos o visualizarlos en forma de mapas.. Dado el código del organismo a través del parámetro %idcode% se conecta a la base de datos en Postgre/PostGIS y es posible acceder a los datos. La forma en que estos datos es devuelta por el servidor de mapas es clasificado por el parámetro classitem lo cual permite definir un estilo personalizado de acuerdo a la clasificación del organismo. Se utilizaron los colores verde, amarillo, rojo y carmelita para los organismos de tipo TD, TS, HU y EX respectivamente. También se utilizó el color de verde claro, naranja y rojo para el radio de los vientos clasificado por intensidad, siendo el radio de los vientos más intensos el de color rojo.

Mapas temáticos

Como hemos visto anteriormente a la infraestructura creada se le puede incorporar cualquier mapa o dato que contenga información geográfica. Ejemplo de estos pueden ser los mapas temáticos creados por el departamento de clima. Pudieran ser mapas de precipitación, mapas de temperaturas, mapas de humedad entre otros. Dichos mapas pueden ser previamente elaborados y posteriormente incorporados a la infraestructura para acceder a ellos en la red local. El único requisito para incorporar estos tipos de mapas es que estén geo-referenciados o que tengan las coordenadas de latitud y longitud para poder geo-referenciarlos.

Por mencionar otro mapa que puede ser incorporado a la infraestructura está el mapa de elevación del terreno. Para incorporar este mapa al sistema fue necesario un pre-procesamiento de los datos.. Estos datos posteriormente fueron procesados con la librería GDAL y se creó el mapa temático de elevación como vemos en el anexo A.6.

Mapas de satélite tiempo real

Otro de los datos que consume el Centro Meteorológico son imágenes de satélite visibles e infrarrojo. Estos datos también pueden incorporarse a la infraestructura solo que estos son externos a la red local y necesitan conexión a internet.

2.2. Requisitos no funcionales

Los requisitos no funcionales describen las propiedades o cualidades que el sistema debe tener. Debe pensarse como propiedades como las características que hacen al producto atractivo, usable, rápido y confiable. Los requisitos no funcionales de este sistema a automatizar son:

Usabilidad

RNF 1 Facilidad de uso

El sistema será fácil de manipular por cualquier persona, incluyendo aquellos que no cuenten con mucho conocimientos informáticos.
El sistema debe ser intuitivo y tener un alto nivel de usabilidad permitiendo que usuarios con nivel básico en la informática, en aproximadamente 30 días puedan explotarlo al máximo.

Fiabilidad

RNF 2 Disponibilidad

El sistema debe estar funcionando en todo momento, solo se detendrá en caso de fallo eléctrico, se reinicie la pc en la que se encuentra montado por motivos de mantenimiento, configuración o cambios significativos en la aplicación.

RNF 3 Disponibilidad de servicios

El sistema debe mantener el servicio en todo momento, permanecerá inactivo o fuera de servicio en caso que se estén realizando tareas de mantenimiento o de configuración.

Restricciones de diseño e implementación

Lenguaje de programación JavaScript.

NetBeans 7.1 como IDE de desarrollo.

Visual Paradigm como herramienta CASE.

Lenguaje de modelado UML 2.0.

Framework ExtJS 3.2.

Interfaz

Interfaces de usuario

La interfaz de usuario debe ser amigable y fácil de entender, para evitar que pueda perderse en la navegación del sistema.

2.3. Requisitos funcionales

Los requisitos funcionales describen lo que el sistema debe hacer, representa capacidades o condiciones que el sistema debe poseer, independientemente de las cualidades o propiedades con las que se relacione. Los requisitos funcionales del sistema a automatizar son:

RF1.1 Buscar información de tormentas y huracanes del atlántico en el periodo de 1851-2012.

RF1.2 Visualizar información de tormentas y huracanes del atlántico en el periodo de 1851-2012.

RF 2.1 Visualizar límites de países en tiempo real.

RF 2.2 Visualizar líneas de costas en tiempo real.

RF 3.1 Visualizar la información de tormentas y huracanes en tiempo real.

RF 4.1 Visualizar capas de datos geográficos de origen regional (núcleos urbanos, hidrografía, municipios, etc.).

RF 5.1 Selección de características físicas.

RF6.1 Visualizar las estaciones meteorológicas de Cuba.

RF6.2 Consultar datos de las estaciones meteorológicas (OBS).

RF 7.1 Mostrar leyenda.

2.4. Modelo de Casos de Uso del Sistema (CUS)

Un modelo es una abstracción del sistema, especificando el sistema modelado desde un cierto punto de vista y en un determinado nivel de abstracción. Una vista, es por ejemplo, una vista de especificación o una vista de diseño del sistema. (Object Management Group)

El modelo de casos de uso del sistema contiene casos de uso y los actores que inicializan los mismos. Los actores del sistema son aquellos individuos u otros sistemas externos que interactúan con el software. El actor establece un rol en el cual varios individuos pueden jugar ese papel. Para el desarrollo de los componentes se identificaron los actores siguientes.

Actor	Descripción
Especialista	Persona encargada de acceder a los datos espaciales y fuentes de datos.

Tabla 1: Descripción Actores del Sistema.

2.4.1. Definición de los casos de usos del sistema.

Luego de haber descrito los actores del sistema se realiza el proceso de identificación de los casos de usos, con ayuda de los requisitos funcionales del sistema, los casos de uso identificados son:

CU 1 Gestionar tormentas y huracanes.

CU 2 Acceder a capas de datos geográficos globales en tiempo real.

CU 4 Visualizar capas de datos geográficos regionales

CU 6 Visualizar estaciones meteorológicas

CU 7 Visualizar mapas de origen global en tiempo real.

CU 8 Mostrar leyenda

Casos de uso extendidos e incluidos

CU 3 Visualizar tormentas y huracanes en tiempo real (extendido).

CU 5 Selección de características.

2.4.2. Diagrama de casos de uso del sistema

El diagrama de casos de uso representa la interacción entre el cliente (Actor) y el sistema que se encuentra en desarrollo, además del orden en que los elementos (casos de uso) interactúan entre sí.

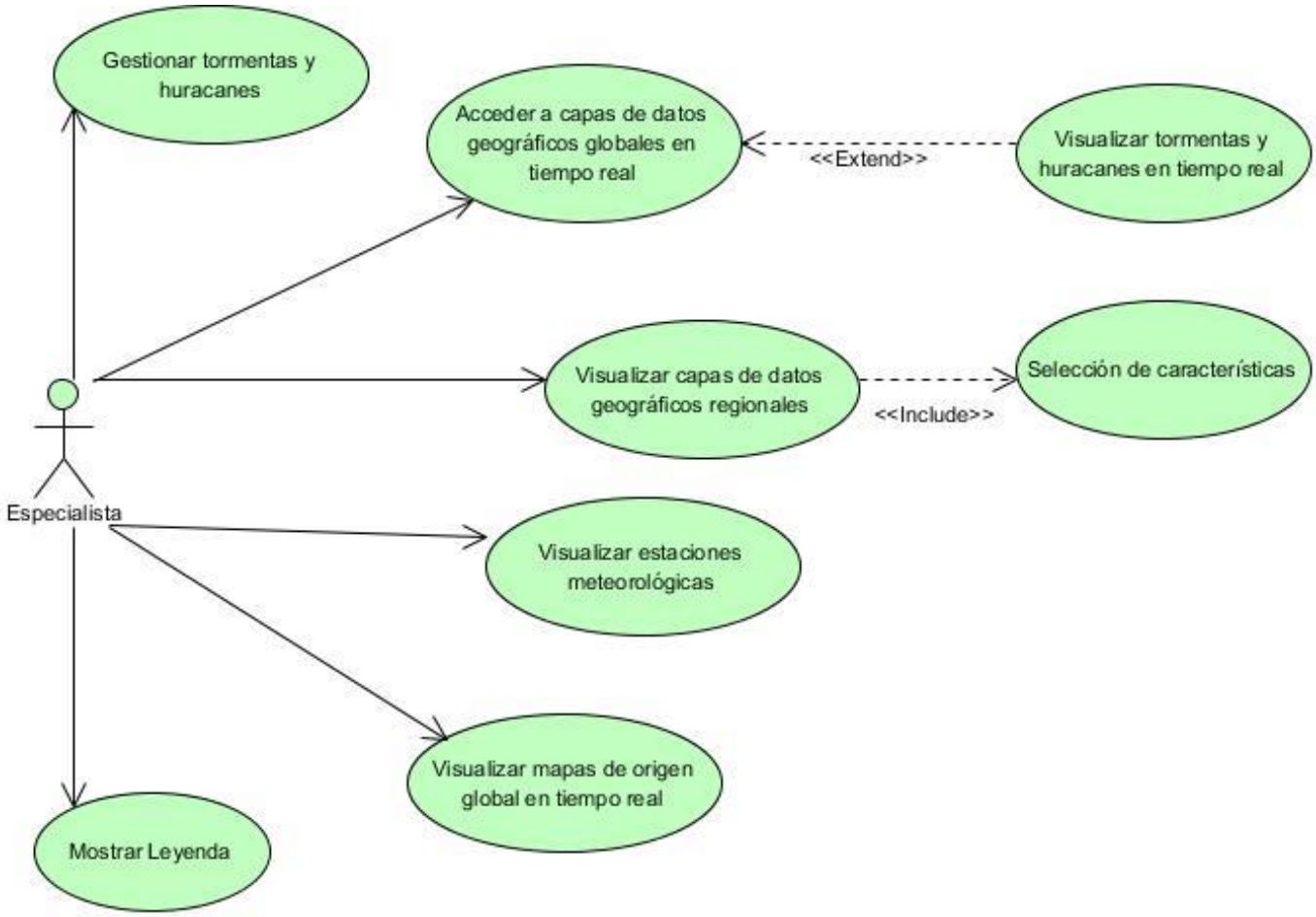


Figura 4 Diagrama de Casos de Uso del Sistema

Descripción del caso de uso Gestionar tormentas y huracanes

<table>
<tr><td>Caso de Uso:</td><td colspan="2">Gestionar tormentas y huracanes</td></tr>
<tr><td>Actores:</td><td colspan="2">Diseñador</td></tr>
<tr><td>Resumen:</td><td colspan="2">El caso de uso se inicia cuando el especialista decide realizar algunas de estas acciones.
Buscar información de tormentas y huracanes: el especialista necesita buscar información referente a eventos pasados entre 1851 al 2012.
Visualizar información de tormentas y huracanes: el especialista necesita trabajar con información referente a eventos pasados entre 1851 al 2012.</td></tr>
<tr><td>Precondiciones</td><td colspan="2"></td></tr>
<tr><td>Referencias</td><td colspan="2">RF 1.1, RF 1.2,</td></tr>
<tr><td>Prioridad</td><td colspan="2"></td></tr>
<tr><td colspan="3">Flujo Normal de Eventos</td></tr>
<tr><td colspan="2">Acción del Actor</td><td>Respuesta del Sistema</td></tr>
<tr><td colspan="2">1. El especialista elige la opción que desea realizar:
• Buscar capa del mapa
• Seleccionar huracán o tormenta</td><td>El sistema muestra :
• El mapa seleccionado en la capa
• Muestra el huracán o tormenta seleccionado</td></tr>
<tr><td colspan="3">Sección “Buscar información de tormentas y huracanes”</td></tr>
<tr><td colspan="2">Acción del Actor</td><td>Respuesta del Sistema</td></tr>
<tr><td colspan="2"></td><td>1. El sistema muestra un mapa</td></tr>
<tr><td colspan="3">Sección “Visualizar información de tormentas y huracanes”</td></tr>
<tr><td colspan="2">Acción del Actor</td><td>Acción del Sistema</td></tr>
<tr><td colspan="2"></td><td>1. El sistema muestra el huracán o tormenta seleccionado</td></tr>
<tr><td colspan="3">Flujos Alternos</td></tr>
<tr><td colspan="2">2a. El diseñador cancela la operación</td><td>2b. El sistema muestra la interfaz anterior.</td></tr>
<tr><td colspan="3">Prototipo de Interfaz</td></tr>
<tr><td>Pos</td><td colspan="2">Mapa mostrado con la trayectoria de los diferentes eventos.</td></tr>
</table>

condiciones	

Tabla 2: Descripción del CU Gestionar tormentas y huracanes

2.4.3. Patrones de casos de uso utilizados

Los patrones de casos de uso ayudan a identificar casos de uso a partir de las características del negocio o de los requisitos del sistema, de una forma más eficiente. Un patrón de casos de uso captura técnicas para que el modelo sea perdurable, reusable y entendible. Estos patrones tienen un enfoque hacia el diseño, y las técnicas son utilizadas en modelos de alta calidad

Para la realización del diagrama de casos de uso del sistema fueron utilizados el patrón CRUD parcial, extensión e inclusión concreta.

El patrón **CRUD** es definido a partir de agrupar determinados requisitos funcionales, que representan las acciones de visualizar y acceder a determinada información, en este caso es evidenciado en los casos de uso gestionar tormentas y huracanes.

El patrón extensión concreta se evidencia en la posible inicialización del caso de uso visualizar tormentas y huracanes en tiempo real, lo cual quiere decir que el especialista decide si ver los diferentes eventos meteorológicos en tiempo real.

Otro de los patrones utilizados fue inclusión concreta, debido a que para visualizar capas de datos geográficos regionales es necesario ver las características de Cuba.

Modelo de diseño

El modelo de diseño es una abstracción de la implementación del sistema. Se utiliza para concebir y para documentar el diseño del sistema de software. Es un producto de trabajo integral y compuesto que abarca todas las clases de diseño, subsistema, paquetes, colaboraciones y las relaciones entre ellos.

2.5. Diagramas de Clases del Diseño

El diagrama de clases del diseño describe gráficamente las especificaciones de las clases de Software y de las interfaces de una aplicación. Normalmente contienen información acerca de las clases, asociaciones, atributos, métodos y dependencias.

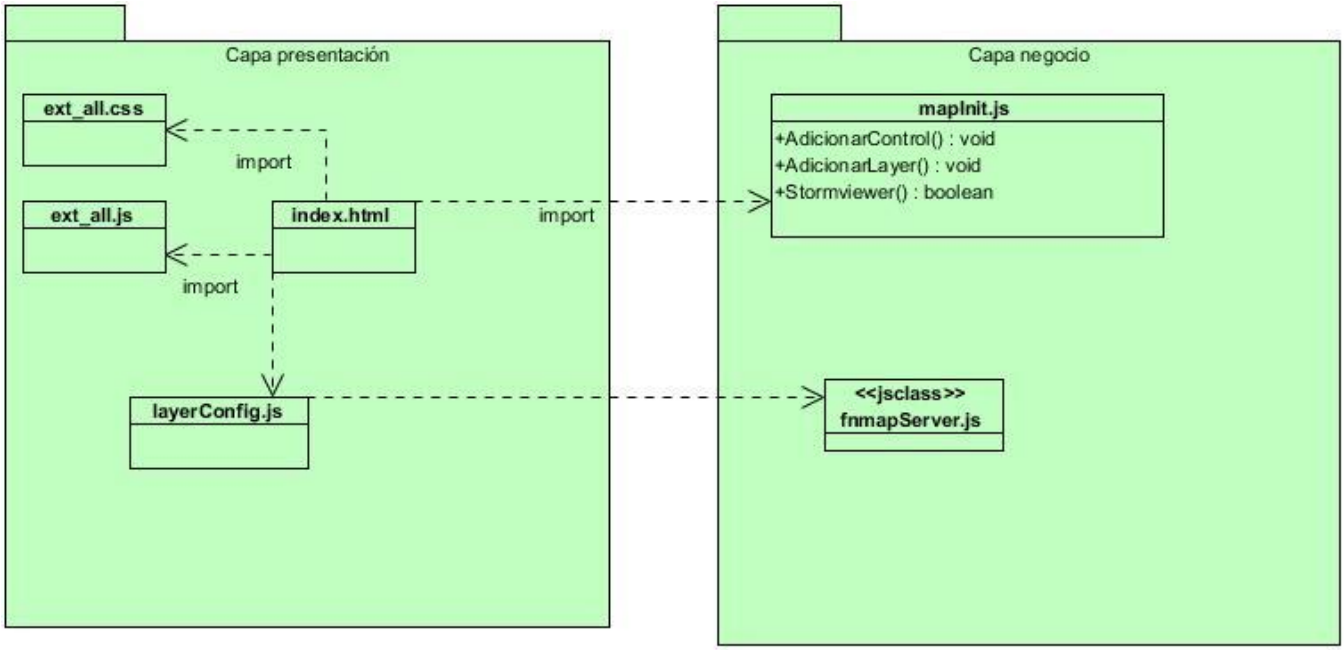


Figura 5 Clases de Diseño del CU Gestionar tormentas y huracanes.

2.6. Diagrama de Secuencia

Los diagramas de secuencias muestran la interacción de un conjunto de objetos en una aplicación a través del tiempo y se modela por cada sección de la descripción de los casos de uso, el diagrama de secuencia contiene los detalles de implementación del escenario, incluyendo los objetos y clases que se usan para implementar el escenario además de los mensajes entre los objetos.

El diagrama de secuencia está constituido por la línea de vida que representa la existencia de un objeto a lo largo de un período de tiempo. El foco de control que es el rectángulo delgado que representa el período de tiempo durante el cual el objeto ejecuta una acción. Para la realización de los casos de usos en el diseño se utilizan los diagramas de secuencia debido a que representan el flujo de acciones con mayor claridad.

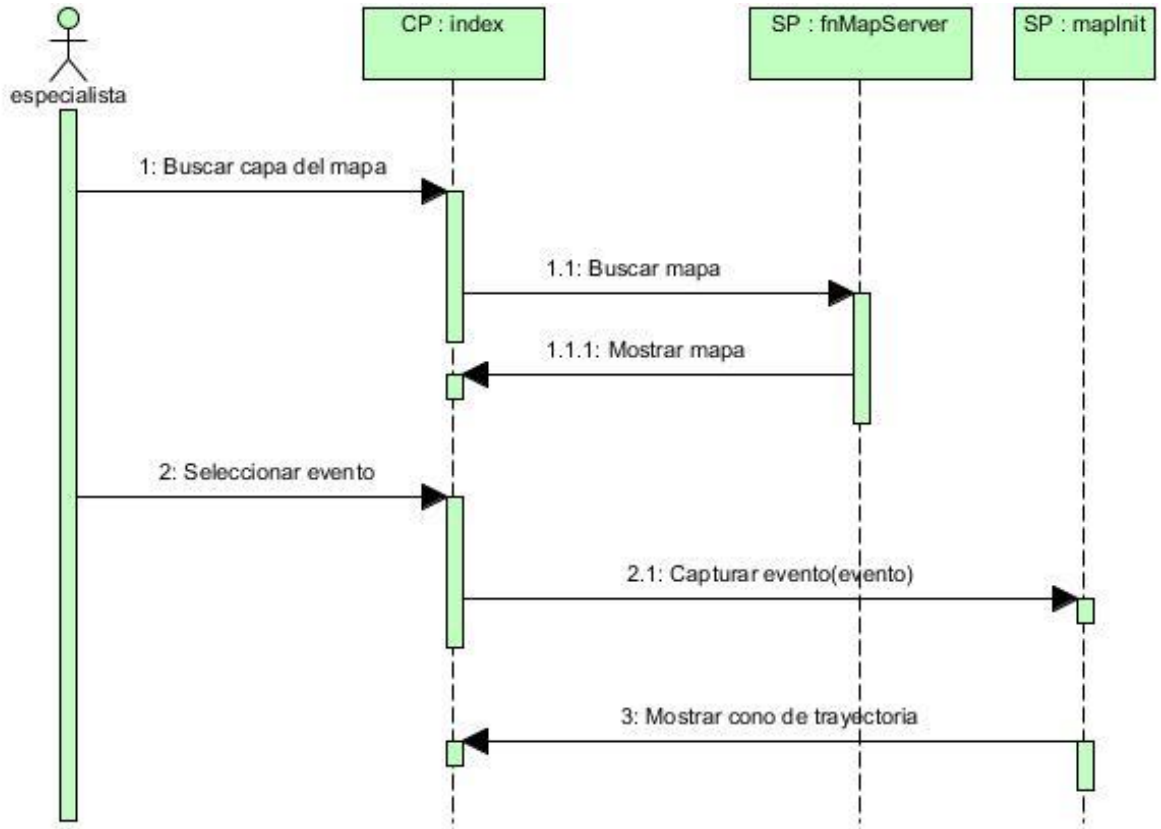


Figura 6 Diagrama de secuencia Gestionar tormentas y huracanes.

2.7. Patrones arquitectónicos utilizados

La arquitectura definida para los productos del centro, es la arquitectura basada en componentes, esta arquitectura identifica a los elementos principales como componentes del software y centra su atención en la integración de varios de ellos para conformar el sistema.

Para la implementación de la aplicación se define la arquitectura por capas. Este patrón genera una organización en cierta jerarquía permitiendo que la capa inferior provea a la capa superior de los servicios que necesite. Para la aplicación se hace uso de la arquitectura de 2 capas.

Capa de presentación:

Esta capa realiza el manejo de las interfaces y la interacción con los usuarios. Es la cara del sistema y su principal deber es lograr una buena comunicación entre el sistema y agentes externos, para ello debe ser completamente funcional.

Capa de negocio (capa lógica)

Esta capa la forman elementos que dentro del sistema se encargan de la automatización de los procesos de negocio realizado por los usuarios. Establece comunicación con la capa superior (capa de presentación), para recibir las solicitudes que son enviadas, y presentar los resultados a la capa superior.

2.8. Patrones de diseño utilizados

GASP (General Responsibility Assignment Software Patterns**)**

- **Alta Cohesión**: En la aplicación cada una de las clases se encuentra enfocada en lo que deben hacer, ya que les han sido asignados los métodos a las clases de forma completa y relacionada, lo que les ayuda a realizar una labor única.

- **Bajo Acoplamiento**: En la aplicación las clases tienen pocas dependencias entre ellas, por lo que existirá menos influencia en los cambios cuando estos se produzcan. En este caso la clase General Panel es la que se comunica con cada una de las clases existentes.
- **Creador:** En la aplicación para acceder a los distintos componentes y obtener las funciones que implementan, se crean instancias de ellos. La ventaja que presenta es que mantiene un bajo acoplamiento entre las clases de la aplicación.

2.9. Modelo de Despliegue

En el diagrama de despliegue se indica la situación física de los componentes lógicos desarrollados. Este modelo es utilizado para capturar los elementos de configuración del procesamiento y las conexiones entre dichos elementos. Cada Hardware es representado como un nodo; un nodo es un elemento donde se ejecutan los componentes, entre ellos existen relaciones que son representados como medios de comunicación tales como HTTP, TCP/IP, etc.

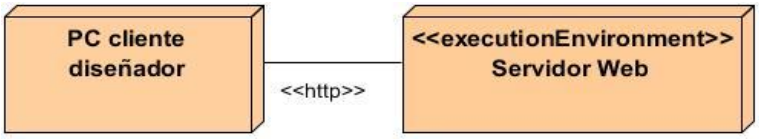


Figura 1 Diagrama de despliegue

Conclusiones del Capítulo

En este capítulo se definió el principal actor que interactúa con el sistema. Se identificaron las funcionalidades que la aplicación debe poseer, así como las propiedades que el cómputo donde se instalará debe tener. Se representaron los requisitos funcionales y no funcionales. Se realizó la descripción del CU gestionar tormentas y huracanes representado en el diagrama de caso de uso. Se identificó el patrón arquitectónico a seguir para la implementación, se mostró el modelo de diseño, con sus clases de diseño y los diagramas de secuencia de una sección de la descripción del caso de uso gestionar tormentas y huracanes.

CAPÍTULO 3: CONSTRUCCIÓN Y VALIDACIÓN

Introducción

Este capítulo se centra en el flujo de trabajo implementación para dar solución a los requisitos especificados. Se expone el artefacto diagrama de componentes, que describe los elementos físicos del sistema y sus relaciones. Además se muestran una serie de código fuente de la construcción de la herramienta y se valida la solución propuesta a través del criterio de especialistas.

3.1. Diagramas de Componentes

El diagrama de componente muestra las organizaciones y dependencias lógicas entre componentes de software, ya sean componentes de código fuente, binarios o ejecutables. Los diagramas de componentes son muy parecidos a los diagramas de casos de uso y son utilizados para modelar la vista estática de un sistema, estos representan las organizaciones y relaciones de dependencias entre los componentes del software.

La siguiente figura es la representación del diagrama de componentes utilizado en la investigación, el diagrama está compuesto por archivos de extensión js, que responden al funcionamiento del lado del cliente con el uso de ExtJS del lado del servidor y algunos archivos php necesarios para la crear el archivo zip. Más adelante se realiza la descripción de cada uno de los archivos representados.

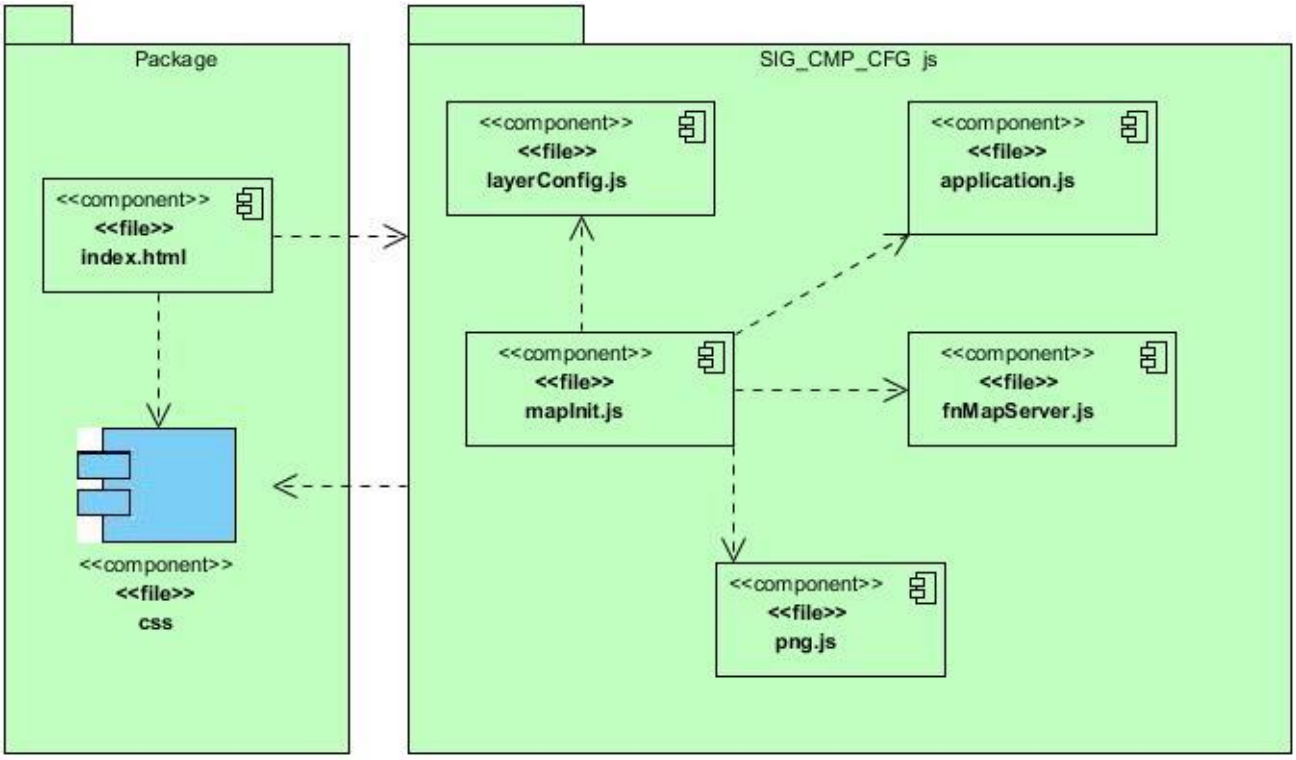


Figura 2 Diagrama de Componente.

Componente	Descripción
layerConfig.js	Carga la configuración general de openlayer.
Aplication.js	Muestra el panel de presentación de los mapas.
index.html	Es la página principal que permite la visualización del sistema web.
Css	Es un componente que permite la visualización de los iconos del sistema web, así como el color azul que presenta la interfaz.
mapInit.js	Es la clase js encargada de inicializar el sistema web.
fnMapServer,js	Es la clase encargada de mostrar el arreglo de mapas guardado previamente.
png.js	Carga imágenes de código en forma de variable.

Tabla 3: Descripción de los componentes definidos en la implementación.

3.2. Estándares de codificación

Los estándares de codificación son por lo general reglas que se deben seguir para realizar la confección del código fuente. La principal ventaja que poseen es que otros programadores pueden entender el código generado, definen buenas prácticas de programación para lograr un código robusto, lo que ayuda en la calidad del software. Muchos lenguajes de programación tiene definidas sus propias reglas de codificación.

Para el desarrollo de la aplicación se tuvieron en cuenta las siguientes reglas:

Estilo de indentación: es una regla que permite delimitar a través de llaves los bloques de códigos lógicos, lo que hace que dicho código sea más legible a los ojos de otros programadores. En la aplicación la indentación va a estar dada por el formato que define el IDE NetBeans, En caso de que una sentencia sea muy larga, se debe cortar la línea y comenzar en la siguiente, después de una coma u operadores lógicos (AND, OR).

Líneas en blanco: En ocasiones se hace necesario establecer visibilidad en el código fuente para mayor organización, las líneas en blanco ayudan a garantizar la estructuración del código, separando las sentencias que pueden pertenecer a clases, bloques, comentarios, etc. Para la implementación de la aplicación se ha decido utilizar esta regla.

Comentarios: Los comentarios ayudan a entender al programador las secciones de código implementado y disminuye el esfuerzo del análisis. En ocasiones se hace necesario explicar que es lo que realiza una función, ya que por intervalos de tiempo el programador olvida que fue lo que pensó en un determinado momento.

Ejemplo de códigos de la aplicación

```
var detectada = navigator.appVersion.split("MSIE") //Primero detectamos que se trata de IE en realidad
var version = parseFloat(detectada[1]) //Luego obtenemos el número de versión

if ((version >= 5.5) && (document.body.filters)) //Preguntamos por la versión 5.5 o superior o si existe el e
{
    for(var i=0; i<document.images.length; i++) // Analizamos todas las imágenes del documento
    {
       var img = document.images[i]; //Tomamos el nombre de la imagen actual en el ciclo
       var imgName = img.src.toLowerCase(); //convertimos a minusculas
      if (imgName.substring(imgName.length-3, imgName.length) == "png") //detectamos la extensión
      {
          var strHTML = "<span " + " style=\"" + "width:" + img.width + "px; height:" + img.height + "px;"
          "filter:progid:DXImageTransform.Microsoft.AlphaImageLoader"
          + "(src=\'" + img.src + "\', sizingMethod='scale');\"></span>" //filter es un elemento único de I
          img.outerHTML = strHTML //Luego obligamos a la imagen a cargar este código en forma de variable
      }
    }
```

Figura 3 Código de cargar imagen

3.3. Validación de la Infraestructura mediante el criterio de especialistas

Las características esenciales de los especialistas seleccionados están dadas por :

- Años de experiencia como especialista en la profesión que se investiga
- Puesto de trabajo actual
- Conocimientos acerca del tema que se investiga
- Años de experiencia como profesional
- Categoría académica
- Otras experiencias laborales
- Creatividad
- Disposición a participar en la investigación
- Capacidad de análisis y de pensamiento

Selección de la competencia de los especialistas

Se seleccionaron 9 especialistas, la selección fue heterogénea y dentro de sus características se destacan las siguientes:

- Todos son especialistas en la profesión que se investiga.
- Todos ocupan cargo actualmente vinculado con la actividad que se investiga.
- Profesionales Universitarios como Geógrafos, Meteorólogos, Informáticos

- El 35 % son Máster
- El 100 % son licenciados o ingenieros
- El mayor número correspondió con especialistas del grupo de pronósticos, especialistas del grupo climatológico e informáticos de la universidad de lo que demuestra el conocimiento de estos acerca del tema, por tratarse la propuesta “Infraestructura de Datos Espaciales en el Centro Meteorológico Provincial”, que es a fin a estas profesiones, por lo que existe mayor comprensión de los objetivos abordados en el mismo.
- Todos tienen información acerca de lo que son datos espaciales e Infraestructura de Datos Espaciales pues en determinado momento de su vida laboral han ejecutado o han sido objeto de verificación en relación con el mismo.

La propuesta de la Infraestructura de Datos Espaciales se le envió a los 9 especialistas y luego de recibir la respuesta de estos, se obtuvo como resultado que el 100 % de ellos están en condiciones de considerarse aptos para el objetivo de la investigación por poseer un alto conocimiento del tema que se propone validar.

3.4. Validación de la Infraestructura que se propone por los especialistas

La validación de la Infraestructura propuesta se hará mediante un cuestionario donde respondan sus opiniones en torno a la “Infraestructura de Datos Espaciales en el Centro Meteorológico de Cienfuegos” sin contacto entre ellos a la hora de dar sus opiniones y emitiendo sus criterios de muy adecuado (MA), bastante adecuado (BA), adecuado (A), poco adecuado (PA) e inadecuado (I), los fundamentos que se proponen son en cuanto a Pertinencia, Viabilidad, Actualidad Científico-Técnica, Congruencia interna de los objetivos, indicadores y encuestas que integran el mismo, así como si contribuye o no a evaluar la Infraestructura de Datos Espaciales en la unidad seleccionada y sus sugerencias al respecto.

Una vez concluido este proceso se procede a analizar los criterios emitidos por cada uno de los especialistas.

Análisis de los resultados del criterio de especialista

Para proceder al análisis de los criterios dados por los diferentes especialistas los autores decidieron partir de la valoración de los aspectos generales que conforman el programa propuesto. La tabulación se realiza por el Método no Estadístico a través del porciento.

Aspecto No 1: Pertinencia.

El 89 % de los especialistas son del criterio que la pertinencia de la “Infraestructura de Datos Espaciales” es muy adecuada y el 11 % restante la considera como bastante adecuada. Todo lo cual lleva a los autores a plantear que según el criterio de la mayoría, la pertinencia de la propuesta es muy adecuada, por el valor social que reviste la misma, pues su objetivo fundamental es obtener los datos espaciales con un mejor aprovechamiento de los recursos informáticos.

Aspecto No 2: Viabilidad.

El 90% de los especialistas consideran la viabilidad como muy adecuada, por lo que afirman que la propuesta reúne los requisitos para ser implementada en entidades análogas y que posean este recurso para el uso correcto de los datos espaciales.

Aspecto No 3: Actualidad científico-técnica de la propuesta.

De los 9 especialistas consultados 8 son del criterio que la actualidad científico-técnica de la propuesta es muy adecuada (89%) y el restante considera que es bastante adecuada; lo que evidencia un alto grado de cientificidad, ya que la infraestructura propuesta está basada en la unificación de recursos informáticos para un mejor manejo y uso de los datos espaciales. Todos coinciden en que debido a la interoperabilidad le permite acceder desde diferentes aplicaciones a la información. Todos están de acuerdo que una infraestructura basada en estándares es una vía adecuada para que los sistemas sean interoperables.

Aspecto No 4: La propuesta contribuye a mejorar el aprovechamiento de Internet.

Los 9 especialistas que representa el 100%, afirman que la propuesta contribuye a mejorar el aprovechamiento de internet, ya que la mayoría de los datos a los que son accedidos se encuentran en la red local y están normalizados.

Aspecto No 5: Congruencia interna de los objetivos que integran la propuesta.

En relación con este aspecto de los 9 especialistas, 8 (89 %) consideran como muy adecuada la congruencia de los objetivos que se proponen en la Infraestructura y el que resta la evalúa de bastante adecuada, lo que indica que la forma utilizada para proponer los mismos es muy adecuada, ya que se encuentran lógicamente estructurados y concatenados entre sí, manteniendo relación con el objeto a evaluar.

Conclusiones del Capítulo

En este capítulo definimos el diagrama de componentes establecido en la aplicación, lo cual muestra una visión de cómo está distribuido físicamente el sistema. Se mostraron los estándares de programación para el código fuente de la aplicación. Se validó la solución propuesta mediante el criterio de especialistas.

CONCLUSIONES GENERALES

Como se puede observar el Centro Meteorológico Provincial de Cienfuegos cuenta con las herramientas necesarias para acceder a los diferentes servicios de datos espaciales tanto en la red local como en internet ya que se creó el Sistema de Información geográfica Web y se le agregaron al cliente las capas necesarias para visualizar los datos espaciales. Fue posible consumir los datos desde varias aplicaciones gracias a la interoperabilidad y una arquitectura basada en servicios. Se obtuvieron mejoras en cuanto al consumo de internet debido a que los datos de los mapas bases son cargados en la red local y solo se consume de internet el servicio WMS que brinda las imágenes de satélite. También hay un consumo nulo de internet para consultar la trayectoria de los organismos y radio de vientos ya que se encuentran normalizados en una base de datos local en la red del Centro Meteorológico Provincial de Cienfuegos.

RECOMENDACIONES

Habiendo cumplido los objetivos trazados con la realización de este trabajo, se sugiere tomar esta propuesta solo como la primera fase de un proyecto mucho más ambicioso y continuar esta investigación. Se recomienda entonces:

- Continuar el estudio de los sistemas de información geográfica.
- Extender su uso a otras universidades o empresas encargadas en el desarrollo de software
- Mantener la concepción base utilizada en el sistema para mantener la interoperabilidad del mismo.
- Se recomienda poder exportar el gráfico realizado como una imagen png o jpeg parar que el usuario no tenga que hacer PrintScreen.
- Distribuir la aplicación a otros Centros Meteorológicos.

REFERENCIAS BIBLIOGRÁFICAS

[1] **Mónica Miranda, "¿Qué son las nuevas tecnologías?," 23-Apr-2014. [Online]. Available: http://monica-miranda.suite101.net/que-son-las-nuevas-tecnologias-a18755. [Accessed: 23-Apr-2014].**

[2] **Delta Desing, "Delta Desing," 09-Sep-2014. [Online]. Available: http://www.ddelta.com.ar/solutions/appWeb.aspx. [Accessed: 09-Sep-2014]**

[3] **P. A. Burrough and R. A. McDonnell, *Principles of geographical information systems.* Oxford: Oxford University Press, 1998.**

[4] **J. E. Harmon and S. J. Anderson, *The Design and Implementation of Geographic Information Systems.* .**

[5] **K. Chang, *Introduction to Geographic Information System*, 4ta ed. McGraw Hill, 2007.**

[6] **C. G. Infante, "Estándares WEB." .**

[7] **Miguel Angel Manso, *Servicios Web para la interoperatibilidad entre programas definidos por el OGC (Open Gis Consortium).***

[8] **SQA, "CBASQA-Desarrollo de Software," 18-Apr-2014. [Online]. Available: http://cbasqa.wordpress.com/2008/09/02/proceso-de-desarrollo-openup/. [Accessed: 18-Apr-2014].**

[9] **Deperu.com, "El diseño," 09-Apr-2014. [Online]. Available: http://www.deperu.com/abc/diseno/269/el-diseno. [Accessed: 09-Apr-2014].**

[10] **"Openup como alternativa metodológica," 09-Apr-2014. [Online]. Available: http://kasyles.blogspot.com/2008/09/openup-como-alternativa-metodologica.html. [Accessed: 09-Apr-2014].**

[11] **Techtastico, "¿Qué es el diseño web?," 09-Sep-2014. [Online]. Available: http://techtastico.com/post/%C2%BFque-es-el-diseno-web/. [Accessed: 09-Sep-2014].**

[12] **Diccionario de la Informática, "Desarrollador." Científico-Técnica, Ciudad de La Habana, 2005.**

[13] **Colectivo de Autores, *Ingeniería del Software*, 4ta ed. 2005.**

[14] **C. Reynoso and N. Kicillof, *Estilos y Patrones en la Estrategia de Arquitectura de Microsoft*, 1st ed. Bueno Aires, 2004.**

[15] C. Larman, *UML y Patrones*. La Habana: Félix Varela, 2004.

[16] E. Hazzard, "Beginner's Guide. Create, optimize, and deploy stunning cross-browser web maps with the OpenLayers JavaScript web-mapping library Object Management Group." .

BIBLIOGRAFÍA

Leopordo, Carlos, "¿Que es el diseño web?," 15-May-2014. [Online]. Available: http://techtastico.com/post/%C2%BFque-es-el-diseno-web/. [Accessed: 15-May-2014].

Mónica Miranda, "¿Qué son las nuevas tecnologías?," 23-Apr-2014. [Online]. Available: http://monica-miranda.suite101.net/que-son-las-nuevas-tecnologias-a18755. [Accessed: 23-Apr-2014].

"Ayuda para OpenUp." 2006.

E. Hazzard, "Beginner's Guide. Create, optimize, and deploy stunning cross-browser web maps with the OpenLayers JavaScript web-mapping library Object Management Group." .

C. Indriago and E. González, "Captura de datos presentacion.," 09-Apr-2014. [Online]. Available: http://www.wiziq.com/tutorial/107690-Captura-de-datos-presentacion. [Accessed: 09-Apr-2014].

SQA, "CBASQA-Desarrollo de Software," 18-Apr-2014. [Online]. Available: http://cbasqa.wordpress.com/2008/09/02/proceso-de-desarrollo-openup/. [Accessed: 18-Apr-2014].

C. A. García, "Curso diseño de bases de datos relacionales," 09-May-2014. [Online]. Available: http://www.mailxmail.com/curso-diseno-base-datos-relacionales/sistemas-bases-datos. [Accessed: 09-May-2014].

Delta Desing, "Delta Desing," 09-Sep-2014. [Online]. Available: http://www.ddelta.com.ar/solutions/appWeb.aspx. [Accessed: 09-Sep-2014].

Diccionario de la Informática, "Desarrollador." Científico-Técnica, Ciudad de La Habana, 2005.

"Diseño Gráfico y web, Tecnología e Internet," 12-May-2014. [Online]. Available: http://profesionalnet.wordpress.com/category/aplicaciones-ria/. [Accessed: 12-May-2014].

Deperu.com, "El diseño," 09-Apr-2014. [Online]. Available: http://www.deperu.com/abc/diseno/269/el-diseno. [Accessed: 09-Apr-2014].

B. Grady, J. Ivar, and J. Rumbaugh, *El Proceso Unificado de Desarrollo de Software*. 2000.

C. Reynoso and N. Kicillof, *Estilos y Patrones en la Estrategia de Arquitectura de Microsoft*. Buenos Aires.

Colectivo de Autores, *Ingeniería del Software*, 4ta ed. 2005.

K. Chang, *Introduction to Geographic Information System*, 4ta ed. McGraw Hill, 2007.

A. Doyle and C. Reed, "Introductión to OGC Web Services v1," May-2001. [Online]. Available: http://ip.opengis.org/ows/010526_OWSWhitepaper.doc. [Accessed: 09-Apr-2014].

OGC, "OGC Web Services 1.2," May-2002. [Online]. Available: http://ip.opengis.org/ows1.2/rfq.html. [Accessed: 09-Sep-2014].

"Openup como alternativa metodológica," 09-Apr-2014. [Online]. Available: http://kasyles.blogspot.com/2008/09/openup-como-alternativa-metodologica.html. [Accessed: 09-Apr-2014].

P.-W. Ng and C. Larman, "Patrones de Casos de Uso," 09-Sep-2014. [Online]. Available: http://www.ecured.cu/index.php/Patrones_de_Casos_de_Uso. [Accessed: 09-Sep-2014].

P. A. Burrough and R. A. McDonnell, *Principles of geographical information systems.* Oxford: Oxford University Press, 1998.

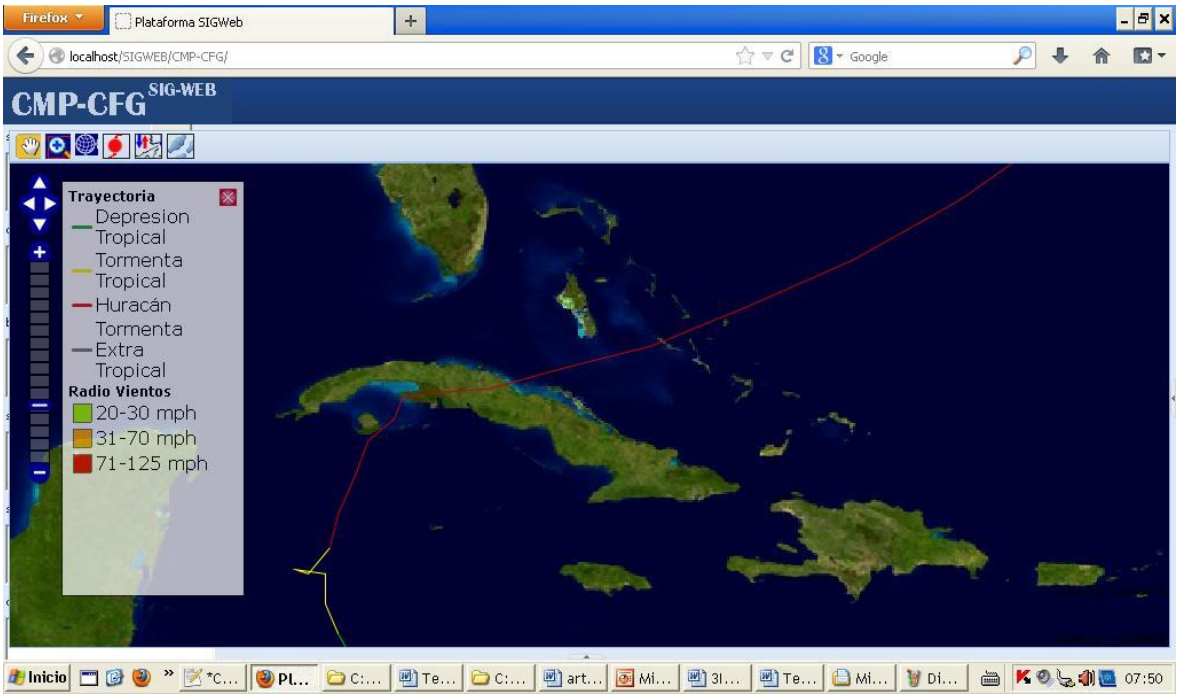
Miguel Angel Manso, *Servicios Web para la interoperatibilidad entre programas definidos por el OGC (Open Gis Consortium).* .

J. E. Harmon and S. J. Anderson, *The Design and Implementation of Geographic Information Systems.*

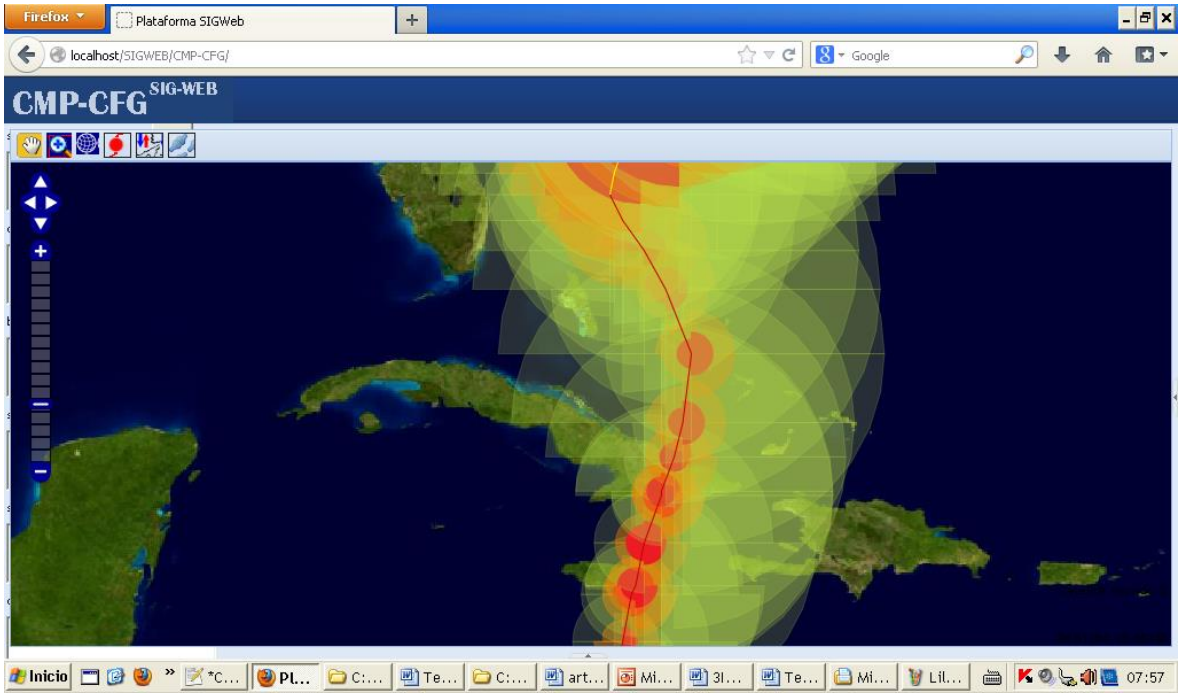
C. Larman, *UML y Patrones*. La Habana: Félix Varela, 2004.

ANEXOS

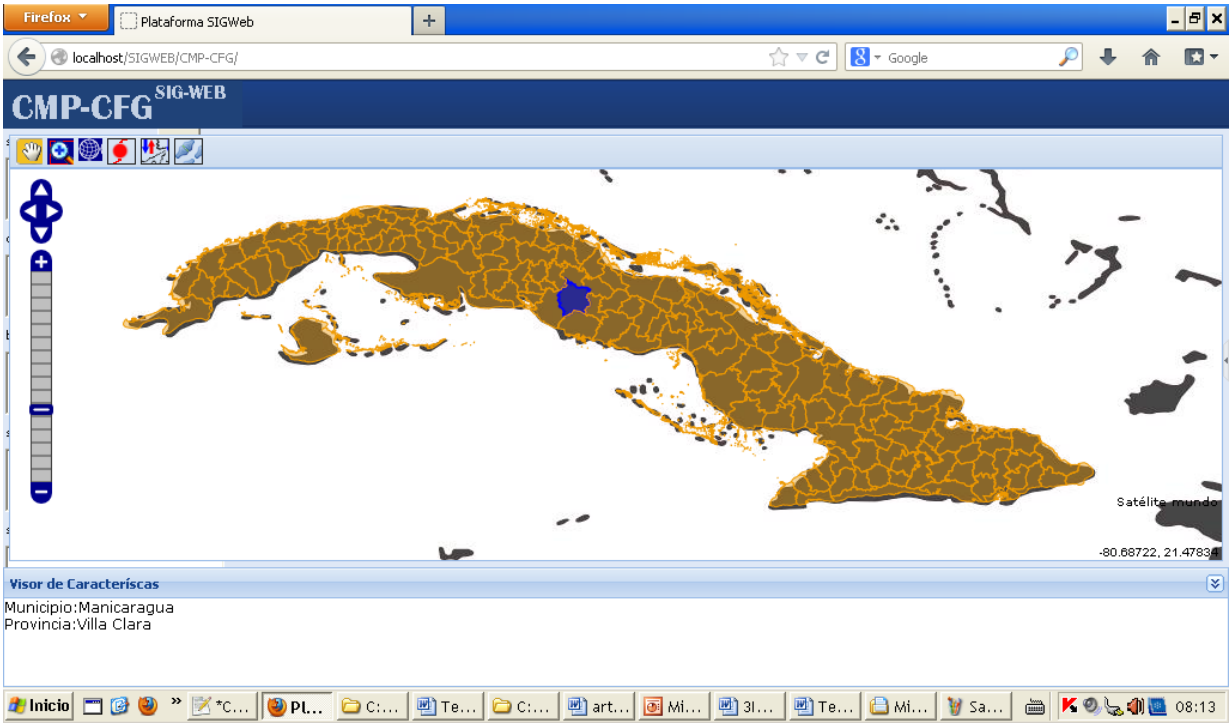
A.1 Mapa personalizado radio de vientos del huracán Lili.

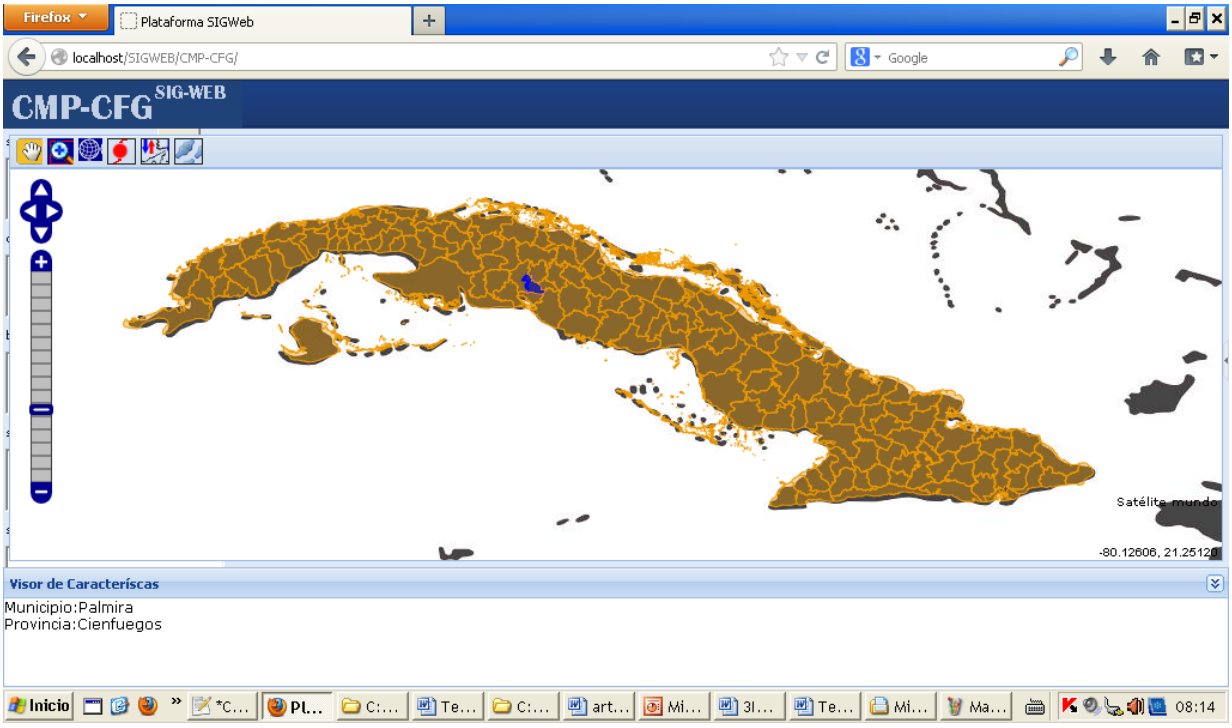


A.2 Mapa personalizado radio de vientos del huracán Sandy.



A.3 y A.4 Características Físicas





A.5 Capa Línea de trayectoria conexión MapServer con Postgre/PostGIS.

```
#POSTGIS Line Postgres Conect
LAYER
CONNECTIONTYPE postgis
NAME "Hurricanesidcode"
VALIDATION
      "idcode" "[A-Z]+[0-9]+"
      'default_idcode' 'AL051935'
END
CONNECTION "user=postgres password=XXXXX dbname=template_postgis_20 host=localhost"
DATA "the_geom from (select * from atltormentas.atlorganismos_lin where atlorganismos_lin.STORMCODE=%idcode%) as subquery using unique gid using srid=4326"
STATUS ON
GROUP "Organismos"
TYPE LINE
CLASSITEM 'stormtype'
CLASS
  NAME 'Baja'
  EXPRESSION "LO"
  Color 0 255 0
END
CLASSITEM 'stormtype'
CLASS
  NAME 'Baja'
  EXPRESSION "TS"
  Color 255 255 0
END
CLASSITEM 'stormtype'
CLASS
  NAME 'Baja'
  EXPRESSION "TD"
  Color 0 200 0
END
CLASSITEM 'stormtype'
CLASS
  NAME 'Baja'
  EXPRESSION "HU"
  Color 200 0 0
END
CLASSITEM 'stormtype'
CLASS
  NAME 'Baja'
  EXPRESSION "EX"
  Color 100 100 100
```

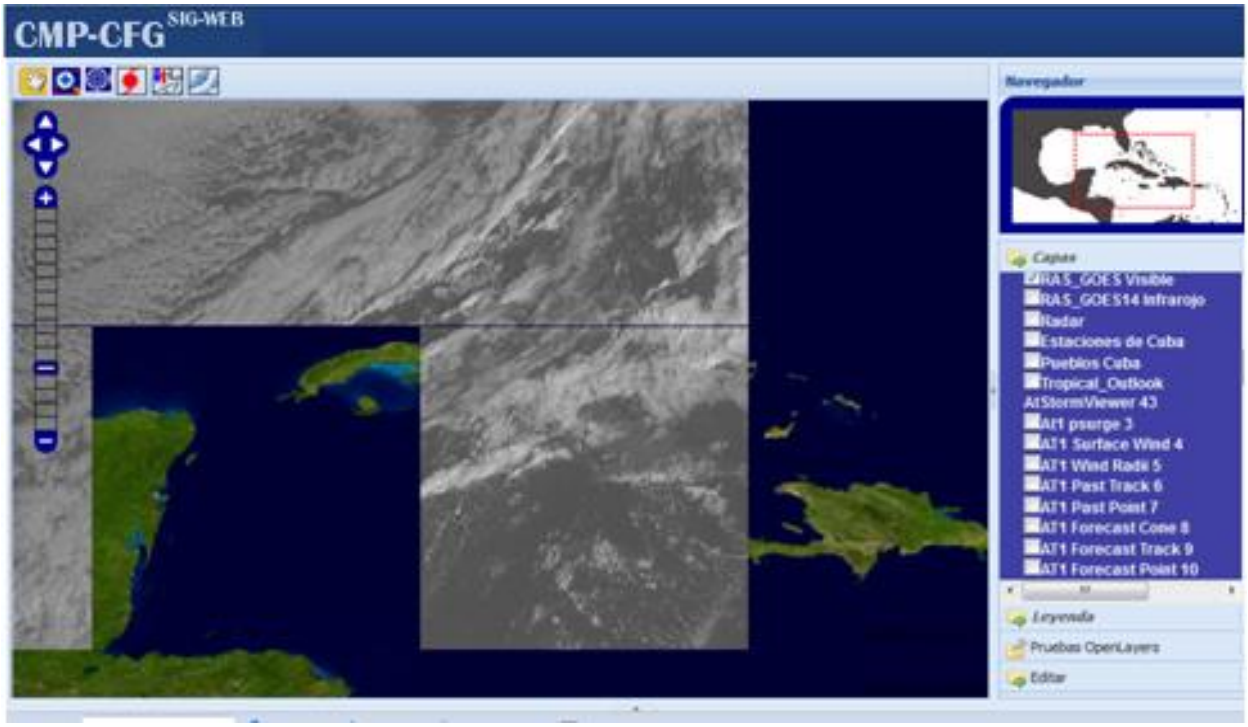
```
END
METADATA
   "gml_include_items" "all" # Consultas
   "gml_geometries" "lineas"
   "gml_lineas_type" "line"
   "wfs_title"  "Organimos por ID"
   "wfs_featureid" "ID"
   "wms_abstract" "Obtener organism dado ID"
   "wms_keywordlist" " Obtener organism dado ID "
   "wfs_enable_request" "*"
   "wfs_srs" "EPSG:4326"
   "wms_srs" "EPSG:4326"
END
END
```

A.6 Capa Línea de trayectoria conexión MapServer con Postgre/PostGIS.

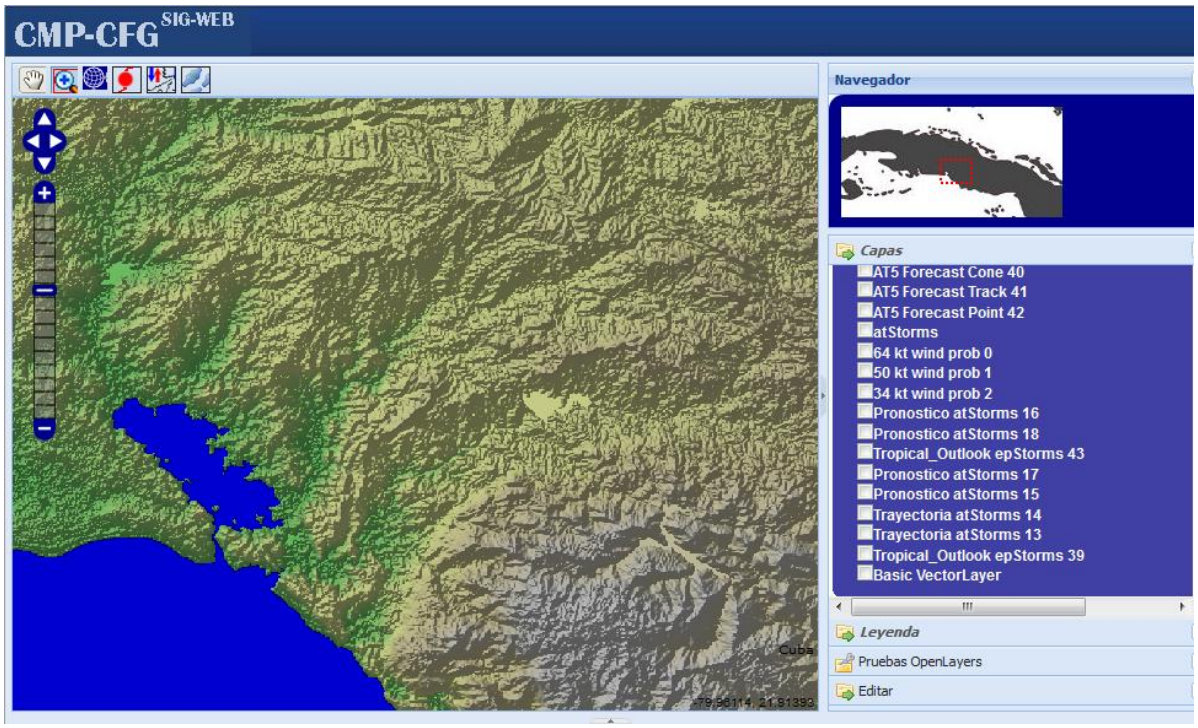
```
LAYER
CONNECTIONTYPE postgis
NAME "HurricaneWindsidcode"
VALIDATION
   "idcode" "[A-Z]+[0-9]+"
   'default_idcode' 'AL051935'
END
CONNECTION "user=postgres password=XXXXX dbname=template_postgis_20 host=localhost"
DATA "the_geom from (select * from atltormentas.atlorganismos_windradii where atltormentas.atlorganismos_windradii.STORMID=%idcode%) as subquery using unique gid using srid=4326"
STATUS ON
GROUP "Organismos"
TYPE polygon
CLASSITEM 'radii'
CLASS
NAME 'Baja'
EXPRESSION "64"
STYLE
  Color 237 28 36
  OUTLINECOLOR 237 28 36
  OPACITY 100
END
END
CLASSITEM 'radii'
CLASS
NAME 'Baja'
EXPRESSION "50"
STYLE
  Color 255 128 0
  OUTLINECOLOR 255 128 0
  OPACITY 40
END
END
CLASSITEM 'radii'
CLASS
NAME 'Baja'
EXPRESSION "34"
STYLE
  COLOR 198 236 83
  OUTLINECOLOR 198 236 83
  OPACITY 20
END
```

```
END
METADATA
   "gml_include_items" "all" # Consultas
   "gml_geometries" "lineas"
   "gml_lineas_type" "line"
   "wfs_title"  "Organimos por ID"
   "wfs_featureid" "ID"
   "wms_abstract" "Obtener organism dado ID"
   "wms_keywordlist" " Obtener organism dado ID "
   "wfs_enable_request" "*"
   "wfs_srs" "EPSG:4326"
   "wms_srs" "EPSG:4326"
END
END
```

A.7Imágenes de satélite



A.8 Mapa temático de elevación



GLOSARIO DE TÉRMINOS

CFG: Cienfuegos.

Ciclo de vida del software: proceso por el cual deben atravesar todos los productos de software: análisis, diseño, implementación, prueba, despliegue y mantenimiento.

CMP: Centro Meteorológico Provincial.

Código fuente: representan instrucciones contenidas en un programa y entendibles por las computadoras.

CU: abreviatura de casos de uso.

Framework: es una estructura de soporte definida en la cual se pueden generar otras aplicaciones a partir de ella, puede incluir herramientas de diseño, lenguajes de programación, bibliotecas de clases y compiladores.

Navegador: Es una aplicación que permite visualizar información que contiene sitios y páginas web.

NOOA: Administración Nacional del Océano y la atmósfera.

NowCoast: Pronóstico de las Costas.

RF: Requisitos funcionales.

RNF: Requisitos nos funcionales.

SIG: Sistema de Información Geográfica.

WFS: Web Features Services Carácterísticas de los mapas.

WMS: Web Map Serrvices, Servicios de Mapas.

GML: Lenguaje de marcado geográfico, sub-lenguaje de XML escrito en una gramática SCHEMA para el modelaje, transporte y almacenamiento de información geográfica.

XML: Lenguaje de marcas extensibles. Metalenguaje extensible de etiquetas desarrolladas por WWW Consortium

1