DESARROLLO E IMPLEMENTACIÓN DE UN PROTOTIPO DE SERVICIO DE CONSULTA GEOREFERENCIADA CON SOPORTE DE COMPUTACIÓN EN LA NUBE

MARIA ALEJANDRA ALFONSO MUÑOZ

CÉSAR AUGUSTO PICO MARTINEZ

UNIVERSIDAD INDUSTRIAL DE SANTANDER

FACULTAD DE INGENIERÍAS FISICOMECÁNICAS

ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA

BUCARAMANGA

2015

DESARROLLO E IMPLEMENTACIÓN DE UN PROTOTIPO DE SERVICIO DE CONSULTA GEOREFERENCIADA CON SOPORTE DE COMPUTACIÓN EN LA NUBE

MARIA ALEJANDRA ALFONSO MUÑOZ
CÉSAR AUGUSTO PICO MARTINEZ

Trabajo de grado para optar al título de Ingeniero de Sistemas

Director:
MSc(c) MANUEL GUILLERMO FLOREZ BECERRA

Co-Director
PhD HOMERO ORTEGA BOADA

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FISICOMECÁNICAS
ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
BUCARAMANGA
2015

A Dios por darme la fuerza, la sabiduría y por estar siempre iluminando mi camino, a mis padres y hermanos por el apoyo incondicional que me han brindado durante toda mi vida, los amo y este triunfo es para ellos, a mi novio por ser mi sostén, mi guía, y siempre alentarme a luchar por lo que quiero, te amo y a mis amigos gracias por ser parte de este aprendizaje.

-Maria Alejandra Alfonso Muñoz

A mis padres, Jorge Pico Camacho (Q.E.P.D.) y Bertha Cecilia Martínez de Pico (Q.E.P.D.), quienes con su apoyo incondicional hicieron posible que hoy lograra llegar a donde estoy, ellos que siempre me guiaron con el mejor de los ejemplos y que hoy desde donde quiera que se encuentren me siguen guiando y protegiendo

-César Augusto Pico Martínez

AGRADECIMIENTOS

Al profesor Homero Ortega Boada y al profesor Manuel Guillermo Flórez por confiar en nosotros para desarrollar este proyecto y por guiarnos en el proceso

Al ingeniero Juan David Rodríguez Ariza ya que sin su apoyo y conocimiento esto no hubiera sido posible, gracias infinitas.

A Oscar Arias, ya que con su colaboración pudimos dar el primer paso en el desarrollo de este proyecto.

A Paola Garrido, Jhonatan Carreño y Alfonso Paez del grupo Conuss por su colaboración y apoyo durante el proceso, gracias sinceras.

.

CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

LISTA DE ANEXOS

RESUMEN

TÍTULO: DESARROLLO E IMPLEMENTACIÓN DE UN PROTOTIPO DE SERVICIO DE CONSULTA GEOREFERENCIADA CON SOPORTE DE COMPUTACIÓN EN LA NUBE[*]

AUTORES: María Alejandra Alfonso Muñoz,
César Augusto Pico Martínez[**]

Palabras claves: web service, sensores virtuales, algoritmo DeJong, computación en la nube.

Dado el camino que está tomando la tecnología de los servicios en la actualidad, este proyecto propone una estrategia para llevar a un entrono web, con apoyo de la arquitectura de nube, desarrollos y avances científicos que en la en este momento se encuentran en un aislamiento informático, siendo por el momento aplicaciones de escritorio.

El propósito principal de este proyecto es demostrar la implementación de un servicio combinando tecnologías existentes con avances innovadores, a través del uso de metodologías agiles como SCRUM, entornos de desarrollo java web, bases de datos *open source*, lenguajes de desarrollo como java, php, javascript y html5 y procesamiento enla nube.

La conversión de un algoritmo de escritorio a un *web service* se lleva acabo mediante un procedimiento que puede ser estandarizado como lo es el siguiente:

1. Analizar el algoritmo e identificar sus procesos principales y entradas de datos para realizarlos.
2. Paramétrizar las entradas de datos según lo requiera el algoritmo.
3. Determinar cuál es el camino a seguir según el lenguaje en el cual este desarrollado el algoritmo.
4. Crear las clases que compondrán el *web service* que se desea crear.
5. Crear la instancia (máquina virtual) en la nube que funcionara como un servidor.
6. Cargar a través de un servidor ftp el ejecutable del *web service* al servidor y finalmente reiniciar el servidor para activar el web service.

Llevar a cabo la conversión a un entorno *web*, con soporte de computo en la nube, de un desarrollo tan importante como lo es el algoritmo de radiopropagación *DeJong*, a través del uso de máquinas virtuales como servidores, representa una contribución importante a las técnicas de Administración del Espectro, proporcionando así una herramienta que logra combinar la simulación con entornos reales, integrando sensores virtuales y la alta disponibilidad que proporciona la nube.

[*] Proyecto de grado

[**] Facultad de Ingenierias fisicomecanicas Escuela de Ingenieria de sistemas e informática Director Manuel Guillermo Florez Becerra. Codirector: Homero Ortega Boada

ABSTRACT

TITLE: DEVELOPMENT AND IMPLEMENTATION OF A PROTOTYPE OF REFERENCED GEOGRAPHIC SERVICE WITH CLOUD COMPUTING SUPPORT*

AUTHORS: María Alejandra Alfonso Muñoz
César Augusto Pico Martínez**

Keywords: web service, virtual sensors, DeJong algorithm, cloud computing.

Given the way that the service technologies are taken in the present, this project proposes a strategy to take, with a cloud architecture support, scientific developments and scientific advances to a web environment.

The main purpose of this project is to demonstrate the implementation of a service combining existing technologies with innovative developments, through the use of agile methodologies like SCRUM, java web development environments, open source databases, development languages like java, php, javascript and html5 and cloud computing.

The conversion of a desktop algorithm to a web service is performed by a method that can be standardized as follows:

1. Analyze the algorithm and identify their main processes and data inputs to perform them.
2. Parameterize the data inputs as required by the algorithm.
3. Determine the way forward according to the language in which the algorithm was developed.
4. Create the classes that make up the web service that want to be created.
5. Create the instance (virtual machine) in the cloud that works as a server.
6. Upload via FTP server the web service executable to the server and eventually reboot the server to activate the web service.

Carrying out the conversion to a web environment, cloud computing supported, such an important as DeJong radio propagation algorithm development, through the use of virtual machines as servers, represents a significant contribution to Spectrum Management techniques, providing a tool that manages to combine the simulation with real environments, integrating virtual sensors and high availability provided by the cloud.

* Project of grade

** Faculty of Physical Ingenierias mechanical engineering school and computer systems Director Manuel Guillermo Florez Becerra. Codirector: Homero Ortega Boada

INTRODUCCIÓN

La mayor preocupación de las grandes empresas hoy en día es tener conectadas a las personas, esto es importante para su éxito, pero además de esto la reciente preocupación gira entorno a la interconexión digital de objetos cotidianos con internet, o más conocido Internet de las cosas (IoT); La empresa de investigación tecnológica Gartner[1] estima que pronto habrá más dispositivos conectados a internet que humanos en el planeta y que incluso hacia el año 2020 cada casa podría tener más de 500 dispositivos conectados. En este tema la Universidad Industrial de Santander tiene varios avances, pero hasta el momento son sensores y equipos físicos conectados a internet por lo que se piensa que con este proyecto de grado se ha llegado a un momento más importante donde se introduce el concepto de sensores virtuales.

Para este proyecto se mostró que a partir de un algoritmo basado en la lógica de un sensor real, se puede construir un sensor virtual; los algoritmos de propagación son dispositivos virtuales que brindan información de medición del espectro, lo cual es clave para unirlos, junto a los equipos de medición dentro del concepto del Internet de las Cosas (IoT) con toda la importancia que representa para la sociedad y la industria.

Se propone entonces una plataforma donde deben convivir los sensores virtuales en este caso el algoritmo de propagación Dejong[2] con los sensores reales, esta definición es algo que la empresa National Instruments denomina el Internet de las

[1] TRENDS T., ACCELERATE T., and PRODUCTIVITY Y., “NI Trend Watch 2015 Technology Trends in 2015 : The Platform is King of the IoT Era,” 2015.
[2] MEJÍA Y. y PINEDA M. Implementación en java de un algoritmo de radiopropagación basado en el modelo de de jong y los requisitos de propagación de los sistemas imt-advanced. 2011.

Cosas Industriales (IIoT)[3], para referirse a cosas complejas que pueden estar conectadas a la Internet como sensores, máquinas de la industria o equipos de medición.

Para la implementación de este modelo de plataforma, se tomó el algoritmo de propagación De Jong[4], al cual se le aplicó una arquitectura de web services tipo REST, computación en la nube por medio del sistema OpenStack y máquinas virtuales, también se implementó el patrón de diseño modelo vista controlador (MVC) por medio del framework Yii, se utilizó la tecnología de mapas con la APIv3 de Google Maps y se realizó la comunicación de los servidores por medio de AJAX, JSON y jQuery para tener finalmente una plataforma de servicios que incluye tanto sensores reales como virtuales con computo en la nube.

Lograr que este algoritmo de propagación se comporte como un sensor virtual facilitara a los usuarios naturales ó entes del estado, la planeación de infraestructura debido a que brindara la posibilidad de predecir y conocer la potencia que emiten las antenas de telecomunicación en ciertas áreas geográficas de interés, sin necesidad de recurrir a una costosa implementación real, otra ventaja sería que al comparar los resultados proporcionados del sensor virtual con el sensor real, sería posible detectar desviaciones suficientemente grandes como para identificar una infracción o uso indebido del espectro.

La necesidad de encontrar métodos que impulsen el desarrollo de aplicaciones y servicios en los que convergen las capacidades de todos esos dispositivos con las tecnologías de la información y las comunicaciones es lo que conduce a la necesidad de identificar las mejores prácticas para el desarrollo de este tipo de soluciones.

[3] TRENDS T., ACCELERATE T. y PRODUCTIVITY Y.. Op. cit.,
[4] MEJÍA Y. y PINEDA M. Op. cit.,

El grupo de Investigación RadioGis de la escuela de Ingeniería eléctrica y electrónica, en colaboración con el Grupo de investigación Conuss de la escuela de ingeniería de sistemas e Informática, buscan demostrar cómo los sistemas de gestión del espectro pueden beneficiarse de un modelo de servicio con aplicaciones en la nube.

1. PRELIMINARES

1.1 PLANTEAMIENTO Y JUSTIFICACIÓN DEL PROBLEMA

En este proyecto se busca implementar un servicio que permita a cualquier persona con la ayuda de cualquier terminal (Smartphone, computador, Tablet, etc), consultar los niveles de señal que reciben desde una antena, tomando en cuenta diversos factores que influencian la radiación. Esto tiene muchas aplicaciones, no solo para ingenieros visionarios, también puede servir como herramienta de apoyo para que la comunidad unida con la academia pueda tomar parte activa en el despliegue de infraestructura de antenas que más le conviene a una ciudad inteligente.

Gracias a esta iniciativa, la UIS tiene la posibilidad de unirse a varios otros actores para atacar una problemática que puede resumirse así: La población y los operadores están enfrentados por causas contradictorias. Por un lado la población exige mayor calidad en los servicios de voz y datos, por el otro lado se opone fuertemente al despliegue de la infraestructura necesaria para ello; el resultado es una infraestructura poco óptima en las ciudades lo cual genera preocupación. Una manera de solucionar el problema con el servicio consiste en crear una red de equipos de medición de la radiación, pero esto resultaría extremadamente costoso, ya que se tendría que comprar gran cantidad de medidores a precios muy elevados, contratar una gran cantidad de personal, asumir costos de transporte, cámaras de seguridad para inspeccionar cualquier anomalía en los equipos, entre otros. Con este razonamiento está plenamente

justificada la necesidad de apostarle al desarrollo de servicios apoyados en modelos de simulación con cómputo en la nube.

Con el apoyo de la Vicerrectoría académica de la UIS, el grupo de investigación RadioGis y el grupo de investigación Conuss se plantea este proyecto de grado para dar solución a esta problemática. Este proyecto enfatiza especialmente en el algoritmo de propagación De Jong por ser de gran perspectiva para los presentes y futuros sistemas de comunicación (4G y 5G).

La principal componente de innovación que este proyecto de grado consiste en involucrar un nuevo tipo de medición, es decir, la medición basada en simulación, con lo cual se introduce un término poco escuchado, “sensor virtual”. Los sensores virtuales y la manera en que se propone su implementación tendrá sin duda un gran impacto en la manera en que se desarrollan y se pone al servicio de la comunidad los sistemas de gestión del espectro; pero también sirve como base para todo tipo de aplicaciones donde existan simuladores para la previsión del comportamiento en diferentes áreas ya sea de utilidad para el agro, la industria, el medio ambiente, la exploración mineral, etc. Además marca el camino a transitar para que ingenieros de diversas profesiones puedan complementarse mutuamente.

1.2 OBJETIVOS

1.2.1 Objetivo general. Desarrollo de un prototipo de servicio de consultas web georeferenciadas para la simulación de radio propagación de la señal, basado en el algoritmo De Jong modificado*.

1.2.2 Objetivos específicos:

- Definir a partir de las tecnologías disponibles, la arquitectura que mejor responda a la necesidad del servicio y que se pueda adaptar fácilmente a servicios similares.
- Implementar un Web Service que puede ser invocado para ser ejecutado de acuerdo a las necesidades de la arquitectura.
- Implementar un servicio prototipo que incluye la arquitectura y la consulta mediante una aplicación primitiva, de fácil acceso para el usuario.
- Valorar el desempeño de la arquitectura a través de pruebas de funcionalidad.
- Formular el modelo a seguir para impulsar el desarrollo de nuevos servicios georeferenciados de señales.
- Implementación en la nube de un prototipo de simulación basado en el algoritmo De Jong proporcionado por el grupo RadioGis.
- Socializar los resultados del proyecto mediante la escritura de un artículo científico.

* En un trabajo previo realizado por el grupo Radiogís se desarrolló una adaptación del modelo De Jong que fue llamado versión modificada del modelo de de Jong, publicado en el I2TS 2010.

2. METODOLOGÍA

Hoy en día se cuenta con innumerables herramientas para la administración de proyectos, debido a que son un factor de éxito muy importante para las organizaciones porque ayudan a garantizar que se logren los objetivos del proyecto en el tiempo previsto y con el presupuesto asignado.

Para el desarrollo de esta tesis de grado se trabajó con la metodología de desarrollo ágil Scrum, esta metodología asume que el proceso de desarrollo de software es impredecible, y lo trata como a una "caja negra" controlada, en vez de manejarlo como un proceso completamente definido. Ésta es una de las principales diferencias entre Scrum y otras metodologías.

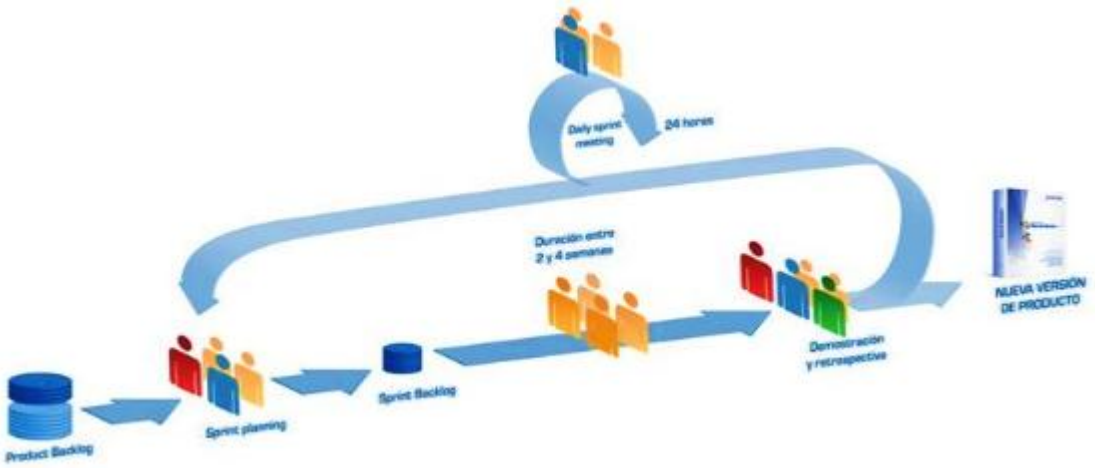
2.1 DESCRIPCIÓN DE LA METODOLOGÍA

El desarrollo se realiza de forma iterativa e incremental. Cada iteración, denominada Sprint, tiene una duración preestablecida de entre 2 y 4 semanas, obteniendo como resultado una versión del software con nuevas prestaciones listas para ser usadas. En cada nuevo Sprint, se va ajustando la funcionalidad ya construida y se añaden nuevas prestaciones priorizándose siempre aquellas que aporten mayor valor de negocio[5].

[5] SOFTENG, "Proceso y Roles de Scrum." [Online]. Available: https://www.softeng.es/es-es/empresa/metodologias-de-trabajo/metodologia-scrum/proceso-roles-de-scrum.html. [Accessed: 13-Oct-2015].

2.1.1 Componentes: Scrum cuenta con un proceso iterativo que se puede observar en la *Figura 1,* cada uno de estos componentes son indispensables para el éxito de un proyecto que se rige bajo esta metodología.

Figura 1. Proceso Scrum



Fuente: SOFTENG Proceso roles de scrum [en línea] disponible en: https://www.softeng.es/es-es/empresa/metodologias-de-trabajo/metodologia-scrum/proceso-roles-de-scrum.html

El proceso de Scrum cuenta con 6 partes en las cuales se analizan los requerimientos, se construyen mediante un control y se hace la presentación de los resultados.

- **Product Backlog:** Conjunto de requisitos denominados *historias*, descritos en un lenguaje no técnico y priorizados por valor de negocio, o lo que es lo mismo. Los requisitos y prioridades se revisan y ajustan durante el curso del proyecto a intervalos regulares.
- **Sprint Planning:** Reunión durante la cual el Product Owner presenta las historias del backlog por orden de prioridad. El equipo determina la cantidad de historias que puede comprometerse a completar en ese sprint, para en una segunda parte de la reunión, decidir y organizar cómo lo va a conseguir.

- **Sprint:** Iteración de duración prefijada durante la cual el equipo trabaja para convertir las historias del Product Backlog a las que se ha comprometido.
- **Sprint Backlog:** Lista de las tareas necesarias para llevar a cabo las historias del sprint.
- **Daily sprint meeting:** Reunión diaria de cómo máximo 15 min. en la que el equipo se sincroniza para trabajar de forma coordinada. Cada miembro comenta que hizo el día anterior, que hará hoy y si hay impedimentos.
- **Demo y retrospectiva:** Reunión que se celebra al final del sprint y en la que el equipo presenta las historias conseguidas mediante una demonstración del producto. Posteriormente, en la retrospectiva, el equipo analiza qué se hizo bien, qué procesos serían mejorables y discute acerca de cómo perfeccionarlos.

2.1.2 Roles Scrum: En Scrum, el equipo se focaliza en construir software de calidad. La gestión de un proyecto Scrum se centra en definir cuáles son las características que debe tener el producto a construir (qué construir, qué no y en qué orden) y en vencer cualquier obstáculo que pudiera entorpecer la tarea del equipo de desarrollo.

El equipo Scrum está formado por los siguientes roles:

- **Scrum master:** Persona que lidera al equipo guiándolo para que cumpla las reglas y procesos de la metodología. Gestiona la reducción de impedimentos del proyecto y trabaja con el Product Owner.
- **Product Owner (PO):** Representa al cliente que usara el software. Se focaliza en la parte de negocio y es el responsable de que el proyecto entregue un valor superior al dinero invertido. Traslada la visión del proyecto al equipo, formaliza las prestaciones en historias a incorporar en el Product Backlog y les asigna un orden por prioridad ajustándolo regularmente.

- **Team:** Grupo de profesionales con los conocimientos técnicos necesarios y que desarrollan el proyecto de manera conjunta llevando a cabo las historias a las que se comprometen al inicio de cada sprint.

2.1.3 Beneficios: Los principales beneficios que proporciona Scrum son:

- Gestión regular de las expectativas del clientey basada en resultados tangibles.
- Flexibilidad y adaptación respecto a las necesidades del cliente, cambios en el mercado, etc.
- Productividad y calidad.
- Alineamiento entre el cliente y el equipo de desarrollo.
- Equipo motivado.

En el mercado hay gran variedad de aplicaciones que están diseñadas para gestionar proyectos con la metodología Scrum, para este proyecto se decidió utilizar una aplicación opensource llamada IceScrum *Figura 2*, ya que tiene una interfaz muy gráfica, cómoda e intuitiva.

Figura 2. Aplicación IceScrum

Fuente: ICESCRUM [en línea] disponible en: https://www.icescrum.com/

IceScrum permite el registro de usuarios y la asignación de roles para cada uno de los miembros del equipo, dependiendo del rol asignado se pueden visualizar y realizar algunas acciones y otras no, por ejemplo el Scrum master es el encargado de crear las tareas que se van a desarrollar para cada Sprint, los miembros del equipo de desarrollo, pueden tomarlas, darles una estimación de tiempo, pero no pueden borrarlas; otra funcionalidad son los gráficos que resumen el comportamiento de los sprint.

Es una herramienta muy completa, que cuenta con todos los componentes necesarios para desarrollar un proyecto bajo la metodología Scrum.

3. MARCO TEÓRICO

3.1 GENERALIDADES DEL ALGORITMO DE JONG RADIOGIS

El algoritmo De Jong con el cual se trabajó en este proyecto de grado, tiene algunos aspectos importantes de estudio para entender su funcionamiento; en este capítulo se describe los inicios del algoritmo, sus componentes, su funcionamiento y las modificaciones e investigaciones en los trabajos de PINEDA, et al[6]. y MELGAREJO, et al.[7], realizados por integrantes del grupo de investigación RadioGis de la UIS.

Yvo Léon Christiaan De Jong elaboró su tesis doctoral con el objetivo de demostrar que era posible crear un modelo de radio propagación determinístico para microceldas con buenos resultados en entornos urbanos, diseñó varios experimentos para identificar los factores más importantes en el modelado del canal inalámbrico[8]. En su tesis quedó demostrado que es importante considerar los emisores bloqueados, cuando no hay línea de vista entre el emisor y el receptor, así como la difracción de la onda con el borde externo de la esquina del edificio y el *scattering*[*], coherente e incoherente, producido por los árboles.

---

[6] PINEDA ALHUCEMA M. D., ORTEGA BOADA H., and NAVARRO CADAVID A., "Modified Version of de Jong Radio Propagation Model," *Int. Informatics Telecommun. Symp. I2TS2010*, pp. 141–147, 2010.

[7] MELGAREJO Y. H. M., BOADA H. O., and ALHUCEMA M. D. P., "A model for urban microcell radio propagation prediction focused on reliable implementation," *2012 IEEE Colomb. Commun. Conf. COLCOM 2012 - Conf. Proc.*, no. 1, 2012.

[8] MEJÍA Y. y PINEDA M. Op. cit.,

[*] En física, proceso en el que la radiación electromagnética o las partículas sufren una deflexión o se propagan en varias direcciones sobre un área determinada.

Más adelante, Pineda et al. desarrollaron un modelo de acuerdo a los estudios realizados por De Jong[9], este modelo, basado en *ray-tracing*[*], se funda en la creación de varios emisores, denominados fuentes virtuales pues en su posición no existe un emisor físico sino uno creado por el algoritmo para simular los efectos de reflexión, difracción o *scattering*, cuyas contribuciones de potencia simultaneas sobre un punto dará como resultado su nivel de señal.

3.1.1 Fuentes virtuales: Son aquellos emisores que no existen físicamente pero que ayudan a modelar el canal simulando los efectos del entorno sobre las ondas electromagnéticas. Este tipo de fuente tiene una serie de parámetros característicos descritos en la Tabla 1, el más importante es el que identifica a la fuente que lo creó, ya sea virtual o real.

Tabla 1. Descripción de los parámetros de las fuentes virtuales

PARÁMETRO	DESCRIPCIÓN
Posición	Ubicación espacial de la fuente.
Señal	Está compuesta por la amplitud de la señal que se emite en Volts, la fase de la señal y la longitud de onda.
Zona de iluminación (ZI)	Es la región en la que tiene validez la fuente, es importante aclarar que las fuentes virtuales irradian en todas las direcciones.
Fuente padre	Es la fuente que dio origen a la fuente virtual.

Fuente: MEJÍA Y. and PINEDA M.

Cabe resaltar que cada fuente virtual puede tener parámetros adicionales dependiendo del fenómeno que represente.

[9] MELGAREJO Y. H. M., BOADA H. O., and ALHUCEMA M. D. P.. Op. cit.,

[*] Algoritmo que consiste en trazar los rayos para determinar las superficies visibles con un proceso de sombreado, que tiene en cuenta efectos globales de iluminación como pueden ser reflexiones, refracciones o sombras arrojadas.

3.1.1.1 Fuentes virtuales de reflexión: De Jong basa su proposición de *ray-tracing* en trabajos previos de RIZK[10] y ATHANASIADOU[11], donde el concepto de fuentes virtuales que resultan de la reflexión ha sido tratado por primera vez. La contribución de De Jong consiste en demostrar la importancia de las interacciones de un rayo no solo con las paredes de los edificios que están frente a la antena transmisora sino también con las paredes que están detrás de estos edificios frontales.

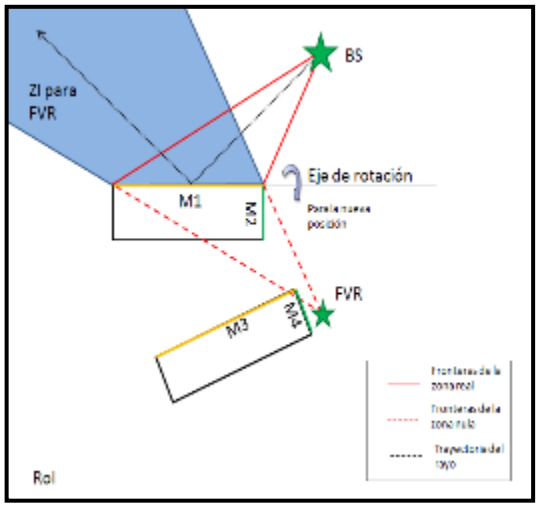
Para identificar la posición de las fuentes virtuales de reflexión se emplea la teoría electromagnética de las imágenes, en este caso, una pared juega el papel de espejo para la onda incidente. En la *Figura 3*, el espejo M1 irradia energía como si una fuente virtual de reflexión (FVR), que se genera a partir del eje de rotación, existiera de manera independiente a la estación base (BS). Un punto de interés (PoI) localizado dentro de la zona de iluminación (ZI) de la fuente virtual de reflexión puede recibir potencia de la FVR y de la BS. Se asume que la potencia de la señal de la FVR es la potencia que llega al punto medio del espejo, sin tener en cuenta las pérdidas por espacio libre, además para cada reflexión se resta un valor constante de 5 dB a la potencia.

La ZI de una FVR es el espacio formado por todos los posibles rayos que alcanzan parte o totalidad del espejo y son reflejados, la ZI empieza en el espejo y finaliza en las fronteras de la región de interés (RoI).

[10] RIZK K., WAGEN J.-F., and GARDIOL F., "Two-dimensional ray-tracing modeling for propagation prediction in microcellular environments," *IEEE Trans. Veh. Technol.*, vol. 46, no. 2, pp. 508–518, May 1997.

[11] ATHANASIADOU G. E., NIX A. R., and MCGEEHAN J. P., "A microcellular ray-tracing propagation model and evaluation of its narrow-band and wide-band predictions," *IEEE J. Sel. Areas Commun.*, vol. 18, no. 3, pp. 322–335, Mar. 2000.

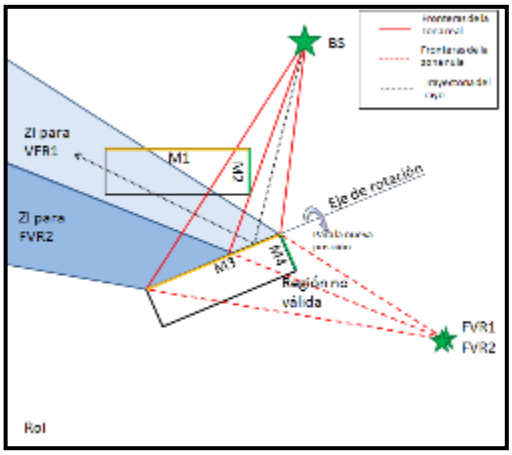
Figura 3. Creación de una fuente virtual de reflexión (FVR)



Fuente: MEJÍA Y. and PINEDA M.

La *Figura 4,* muestra un caso de estudio con cuatro diferentes espejos, de M1 a M4, correspondientes a las paredes de los edificios localizadas en frente de la BS. Antes de de Jong sólo se tenían en cuenta las paredes que no tuvieran obstáculos en frente, es decir el espejo M3 generaría una única FVR. El hecho de que el espejo M3 genere dos fuentes virtuales de reflexión corresponde a un caso especial, que toma en cuenta la transmisión por edificios. El espejo M3 recibe dos potencias diferentes, realmente la señal recibida para la FVR2 además de tener pérdidas por espacio libre presenta las pérdidas por cruzar los edificios. Por esta razón aparecen dos FVR en la misma posición, pero son diferentes en ZI y potencia.

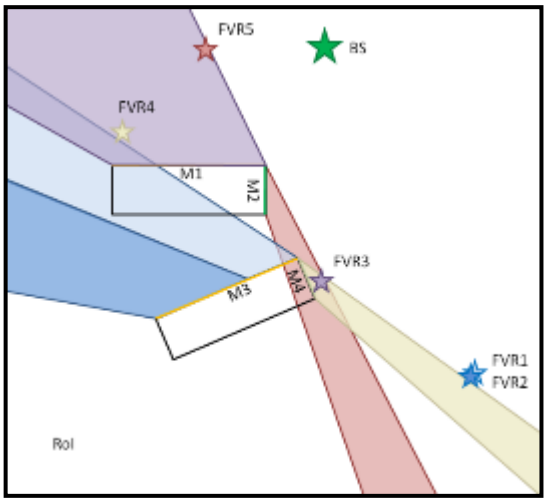
Figura 4. Creación de dos FVR cuando hay transmisión por edificios



Fuente: MEJÍA Y. and PINEDA M.

Por último, si se toman en cuenta los espejos M2 y M3, se obtienen todas las FVR para la RoI como se muestra en la *Figura 5*.

Figura 5. Todas las FVR resultantes de una BS

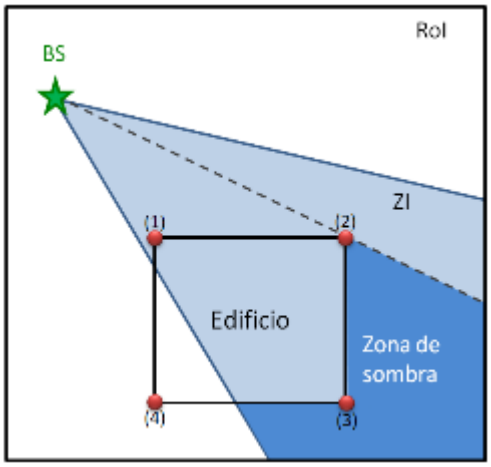


Fuente: MEJÍA Y. and PINEDA M.

3.1.1.2 Fuentes virtuales de difracción (FVD): Los edificios son los elementos que más afectan a las ondas de radio en las zonas urbanas, cuando sus alturas

son muy superiores a las de los transmisores en sus esquinas se presenta el fenómeno de difracción. Desde una vista superior del entorno, las esquinas se verían como cuñas. La difracción es relevante cuando se encuentra en las zonas no iluminadas, o zonas de sombra, del edificio. Tal y como se muestra en la *Figura 6.*

Figura 6. Escenario para la difracción



Fuente: MEJÍA Y. and PINEDA M.

Las fuentes virtuales de difracción se crean a partir de las esquinas de los edificios. Sin embargo, no todas las esquinas de un edificio pueden ser fuentes virtuales de difracción, sólo aquellas que limitan con la zona de sombra y están el línea de vista con la estación base (BS) pueden ser fuentes de difracción. Un ejemplo de análisis de esquinas para determinar las FVDs de la *Figura 6* es descrito a continuación:

- La esquina (1) no puede ser FVD porque está completamente en línea de vista con la BS y no está cerca de ninguna zona de sombra.
- La esquina (2) si es una FVD porque está en línea de vista y limita con la zona de sombra y la línea que va de (1) a (2) sería su pared relacionada.
- La esquina (3) no puede ser FVD porque no tiene línea de vista.

- La esquina (4) no puede ser FVD porque está por fuera de la zona de iluminación de la BS.

Luego de determinar la posición de la FVD es necesario abstraer sus atributos. En la Tabla 2, se muestran los parámetros adicionales que necesita esta fuente.

Tabla 2. Descripción de los parámetros adicionales para la FVD

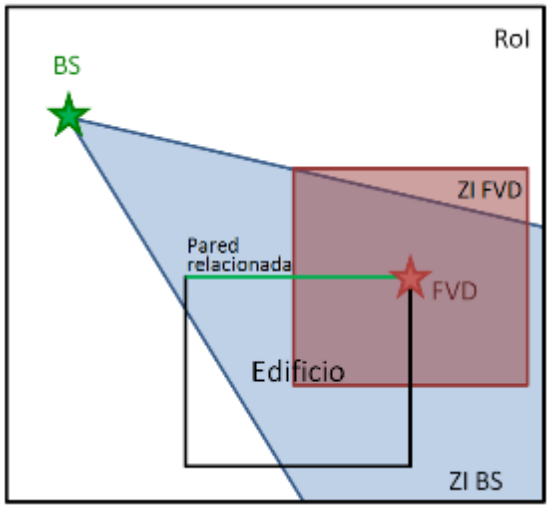
Parámetro	Descripción
Edificio asociado	Edificio sobre el que sucedió la difracción
Pared relacionada	Pared del edificio que fue iluminada durante la difracción.

Fuente: MEJÍA Y. and PINEDA M.

Es necesario calcular los parámetros de la FVD, primero, para la señal se usa el canal virtual (será definido más adelante) para calcular la amplitud y fase de la señal desde la fuente padre hasta la posición de la esquina. La zona de iluminación es un cuadrado cuyos lados tienen una longitud que es directamente proporcional a la distancia que recorrería la onda con la potencia en la esquina hasta alcanzar un umbral preestablecido de -80 dBm[*]. En la *Figura 7* se muestra el resultado visual de una FVD con su respectiva zona de iluminación (ZI FVD) y pared iluminada.

[*] Umbral típico para las comunicaciones móviles inalámbricas, sin embargo depende de la precisión del equipo receptor.

Figura 7. Resultado visual de una FVD



Fuente: MEJÍA Y. and PINEDA M.

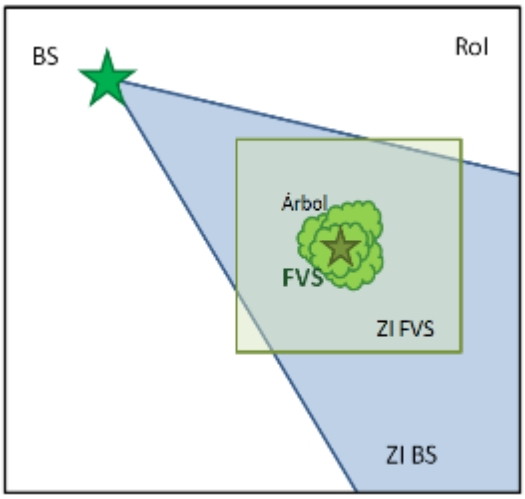
3.1.1.3 Fuentes virtuales de scattering (FVS): En los entornos urbanos, típicos para microceldas, la cantidad de árboles presentes es considerable y su efecto sobre las ondas de radio no debe ser despreciado, especialmente en las frecuencias empleadas para telefonía celular, entre 0.8 y 2.2 GHz, donde la longitud de onda es similar a las dimensiones de las hojas y ramas, o incluso es menor. Teniendo en cuenta que cada árbol afectará directamente al canal es posible crear un emisor en la posición del árbol que irradie de forma omnidireccional el componente incoherente del *scattering*, como explica JONG[12].

En este orden de ideas, cada vez que un emisor o una fuente irradien un árbol, en la posición del árbol se creará una fuente virtual de *scattering*, (FVS). La potencia de una FVS se obtiene a partir del valor de atenuación incoherente causado por el árbol, usando las ecuaciones tensoriales descritas[13], multiplicado por la potencia que llega a su posición desde la fuente o emisor que irradia al árbol. Por otro lado,

[12] JONG Y. Y., "Measurement and modelling of radiowave propagation in urban microcells," 2001.
[13] DEJONG Y. L. C. and HERBEN M. H. A. J., "A Tree-Scattering Model for Improved Propagation Prediction in Urban Microcells," *IEEE Trans. Veh. Technol.*, vol. 53, no. 2, pp. 503–513, Mar. 2004.

la zona de iluminación se obtiene de la misma forma que se hace para las fuentes virtuales de difracción. En la *Figura 8,* se muestra el resultado visual de una FVS con su respectiva zona de iluminación (ZI FVS).

Figura 8. Resultado visual de una FVS



Fuente: MEJÍA Y. and PINEDA M.

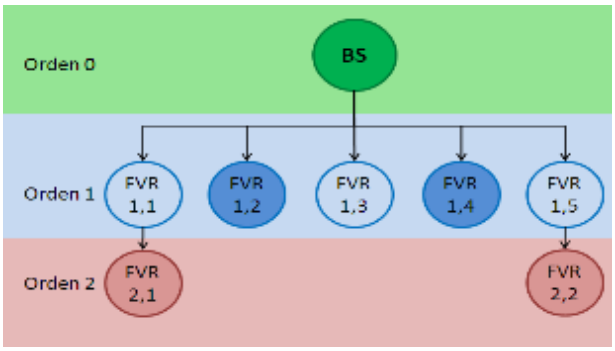
3.1.2 Orden de descendencia: A pesar de que el *ray-tracing* ha sido usado desde hace mucho tiempo sus ventajas no habían sido debidamente implementadas pues se caía en el dilema de exactitud contra tiempo de cómputo. Yvo de Jong, en su tesis de doctorado, estableció la importancia que tiene considerar más de dos reflexiones por trayectoria mostrando que los resultados en la predicción mejoraban significativamente con respecto a los modelos anteriores que sólo consideraban una sola reflexión por trayectoria.

Para De Jong, considerar más de dos reflexiones significa generar más fuentes virtuales de reflexión. Una forma de explicar la manera en que las fuentes virtuales son generadas es mediante un árbol de fuentes*, que funciona de la misma forma

* Árbol de fuentes (*tree sources*): Es un concepto establecido por De Jong en su tesis de doctorado

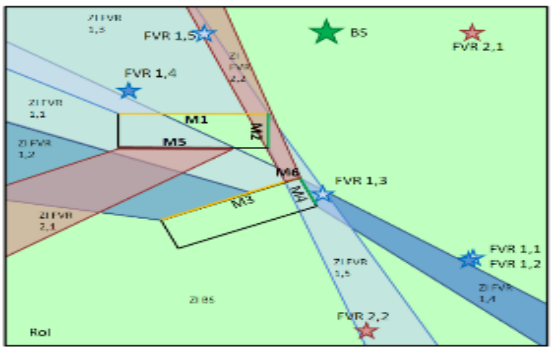
en que lo hace un árbol genealógico. Un ejemplo de árbol de fuentes es el mostrado en la obtenido a partir del mapa ilustrado en la.

Figura 9. Ejemplo de árbol de descendencia para FVR



Fuente: MEJÍA Y. and PINEDA M.

Figura 10. Entorno de muestra para el análisis de la reflexión



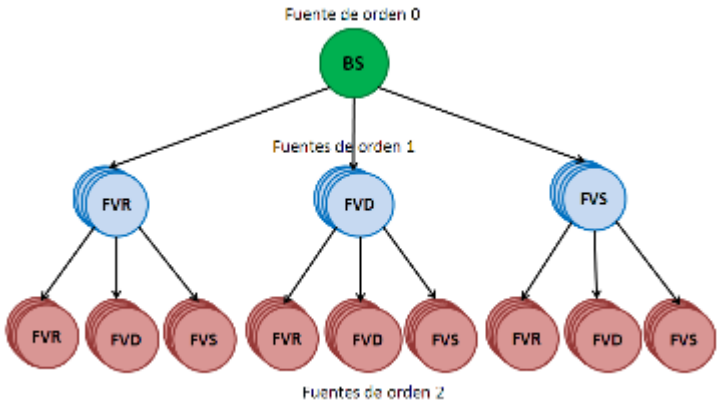
Fuente: MEJÍA Y. and PINEDA M.

Las fuentes virtuales de reflexión fueron creadas, siguiendo el proceso explicado anteriormente, El orden definido para las fuentes, dado en la *Figura 9*, se obtiene mediante el concepto de descendencia sobre las fuentes virtuales creadas. En ese orden de ideas, la BS es la fuente padre y las FVR de color azul son sus hijas. Todas las fuentes reales, en este caso la BS, son fuentes de orden cero, por consiguiente sus fuentes hijas serán de orden uno. Nótese también que no todas las fuentes hijas pueden tener hijos depende de la geometría del entorno, en este

caso la cantidad de fuentes de orden 2 es menor que la cantidad de fuentes de orden 1.

El concepto de descendencia se aplica en general para todas las fuentes virtuales indiscriminadamente. Un ejemplo de árbol de descendencia para todas las fuentes virtuales se muestra en la, considerando que todas las fuentes tendrán fuentes virtuales hijas de todo tipo, reflexión, difracción y scattering.

Figura 11. Árbol de fuentes general (hasta fuentes de orden 2)



Fuente: MEJÍA Y. and PINEDA M.

En la *Figura 11*, se muestran círculos superpuestos por cada FVR, FVD y FVS indicando que más de una fuente virtual de cada tipo puede ser creada, además se muestra que sin importar el tipo de fuente padre sus fuentes hijas pueden ser de cualquiera de los tres tipos ya definidos. Es importante tener en cuenta que antes de avanzar en el orden de las fuentes hijas hay que finalizar completamente con las fuentes del orden actual, es decir completar la determinación de absolutamente todas las FVRs, FVDs y FVSs del orden actual. Trabajos previos que usan el concepto de fuentes virtuales para modelar los fenómenos en la radio

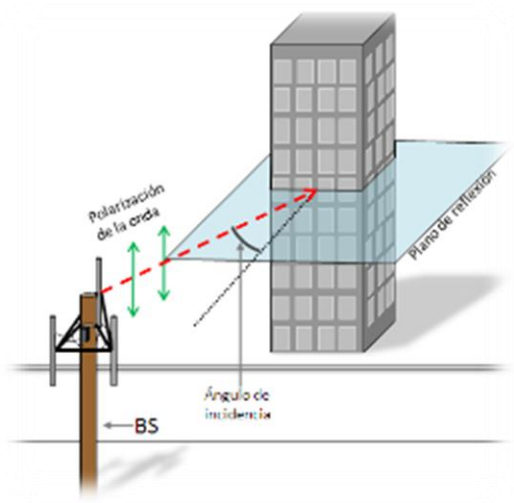
propagación no tienen definido un límite de creación[14], una nueva tendencia es detener la generación cuando se alcance un umbral, esto implica una notable reducción en el número de fuentes creadas.

3.1.3 Reflexión con los edificios: Aunque un coeficiente de atenuación constante por cada rebote brindaba buenos resultados en el modelo planteado por PINEDA, et. al[15], es importante tener en cuenta que el coeficiente de reflexión depende del ángulo de incidencia del rayo sobre la superficie, además, si se quiere ser un poco más cuidadoso con los detalles de implementación, los edificios tienen la capacidad de dispersar energía en varias direcciones dependiendo de la escabrosidad de la superficie y ángulo de incidencia.

Sin embargo, al implementar un coeficiente de reflexión que dependa del ángulo de incidencia hay que tener en cuenta que la onda se propaga vectorialmente, dependiendo de la polarización de la antena, y que la reflexión con un obstáculo cambia su dirección de forma también vectorial, véase la *Figura 12*.

[14] JONG Y. Y.. Op. Cit.,
[15] PINEDA ALHUCEMA M. D., ORTEGA BOADA H., and NAVARRO CADAVID A.. Op. Cit.,

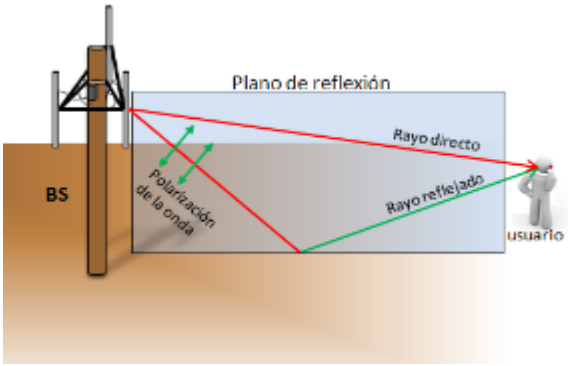
Figura 12. Representación de la reflexión en los edificios



Fuente: MEJÍA Y. and PINEDA M.

3.1.4 Reflexión con la superficie terrestre: Cuando las antenas están muy cerca de la tierra, por debajo de los 10 m, la superficie terrestre afecta a la onda directamente mediante la reflexión[16]. En la se muestra porqué la reflexión con el plano terrestre afecta a la intensidad de la onda recibida.

Figura 13. Reflexión con la tierra



Fuente: MEJÍA Y. and PINEDA M.

[16] MELGAREJO Y. H. M., BOADA H. O., and ALHUCEMA M. D. P.. Op. cit.,

Son muchos los detalles que se requieren para modelar adecuadamente este efecto, sin embargo, para entornos microcelulares es posible considerar que la tierra, en su área de cobertura, es plana manteniendo como fenómeno predominante, sobre la onda, a la reflexión.

3.2 LA NUBE

Figura 14. La nube, una Utilidad para todos

Fuente CREATIVAWEB servicios en la nube [en línea] disponible en: http://www.creativaweb.co/servicios-en-la-nube-para-pymes/

La computación en la nube (Cloud Computing), es un sistema informático basado en internet, máquinas virtuales y centros de datos remotos para la gestión de servicios de información y aplicaciones que permite gestionar archivos y utilizar aplicaciones sin necesidad de instalarlas en la computadora, esto se ha convertido en un nuevo paradigma tecnológico de gran impacto social[17].

[17] MILLER M., *Cloud Computing: Web-Based Applications That Change the Way You Work and Collaborate Online*. 2008.

Con los avances significativos en Tecnología de Información y Comunicaciones (TIC) en el último medio siglo, se piensa que la computación llegara a ser la quinta utilidad (después del agua, la electricidad, el gas y la telefonía). Esta utilidad, como los otros cuatro servicios públicos existentes, proporcionará el nivel básico para satisfacer las necesidades cotidianas de la comunidad en general[18].

La implantación y el mantenimiento de una infraestructura de TI pueden suponer un reto cuando al mismo tiempo se ha de dar respuesta a las necesidades de un entorno empresarial cambiante y de unos equipos de desarrollo de aplicaciones altamente dinámicos. Los gastos de capital iniciales y los costes operativos son elevados y a menudo las infraestructuras se infrautilizan y resulta difícil su reasignación a otras aplicaciones y proyectos. Las arduas tareas de adquisición, configuración y mantenimiento de los entornos de hardware y software suelen efectuarse de forma manual, lo que se traduce en unos altos costes de mano de obra y en un mayor riesgo de que se produzcan errores debidos a configuraciones inadecuadas.

Por último, es necesario contar con una solución segura que dé soporte a usuarios finales y equipos de trabajo, con independencia de si se encuentran ubicados en la misma ciudad o en cualquier punto del planeta. La computación en la nube puede ayudarle a hacer frente a estos retos; asimismo, puede ayudarle a responder a unas cargas de trabajo impredecibles y a reducir los ciclos de desarrollo y despliegue de aplicaciones[19].

La Nube está propiciando una nueva revolución industrial soportada en las fábricas de «datos» (Centros de Datos, Data Centers) y fábricas de «aplicaciones

[18] BUYYA R., YEO C. S., VENUGOPAL S., BROBERG J., and BRANDIC I., “Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility,” *Futur. Gener. Comput. Syst.*, vol. 25, no. 6, pp. 599–616, Jun. 2009.
[19] GLOBAL I. B. M. and SERVICES T., “máquinas virtuales en la nube de IBM,” pp. 1–4.

Web» (Web Apps). Esta nueva revolución producirá un gran cambio social, económico y tecnológico, pero al contrario que otras revoluciones será «silenciosa» al igual que lo ha sido la implantación Internet y la Web en la Sociedad.

3.2.1 Definiciones de la computación en la nube: Esta definiciones están basadas según el informe "The NIST Definition of Cloud Computing" del instituto nacional de estándares y tecnología (NIST)[20], donde dice que la computación en la nube se compone de cinco características esenciales, tres modelos de servicio, y cuatro modelos de implementación, ver *Figura 15.* También están basadas en el libro Cloud Computing. Retos y Oportunidades[21] que describe las soluciones de *cloud computing* disponibles en el mercado en la actualidad y cuenta que estas soluciones admiten diferentes clasificaciones según el aspecto que se tenga en cuenta para realizar dicha clasificación.

3.2.1.1 Características esenciales:

1. Autoservicio bajo demanda, esta característica permite al usuario acceder de manera flexible a las capacidades de computación en la nube de forma automática a medida que las vaya requiriendo, sin necesidad de una interacción humana con su proveedor o proveedores de servicios cloud[22].
2. Acceso sin restricciones, característica consistente en la posibilidad ofrecida a los usuarios de acceder a los servicios contratados de cloud computing en

[20] P. Mell and T. Grance, "The NIST Definition of Cloud Computing Recommendations of the National Institute of Standards and Technology," *Nist Spec. Publ.*, vol. 145, p. 7, 2011.
[21] E. V. Alberto Urueña, Annie Ferrari, David Blanco, *Cloud Computing. Retos y Oportunidades*. España, 2012.
[22] P. Mell and T. Grance. Op. cit.,

cualquier lugar, en cualquier momento y con cualquier dispositivo que disponga de conexión a redes de servicio IP[23].

3. Multiusuario, capacidad que otorga el cloud, que permite compartir los medios y recursos informáticos (almacenamiento, memoria, ancho de banda, capacidad de procesamiento, máquinas virtuales, etc.) de los proveedores hacia múltiples usuarios, las capacidades se van asignando de forma dinámica según sus peticiones. permitiendo la optimización de su uso[24].
4. Agilidad en la escalabilidad, característica o capacidad consistente en aumentar o disminuir las funcionalidades ofrecidas al cliente, en función de sus necesidades puntuales sin necesidad de nuevos contratos ni penalizaciones. Esta característica, relacionada con el —pago por uso, evita los riesgos inherentes de un posible mal dimensionamiento inicial en el consumo o en la necesidad de recursos[25].
5. Abstracción, característica o capacidad de aislar los recursos informáticos contratados al proveedor de servicios cloud de los equipos informáticos del cliente. Esto se consigue gracias a la virtualización[26], con lo que la organización usuaria no requiere de personal dedicado al mantenimiento de la infraestructura, actualización de sistemas, pruebas y demás tareas asociadas que quedan del lado del servicio contratado.

3.2.1.2 Modelos de servicio Cloud:

- **Infrastructure as a Service (IaaS):** Este modelo consiste en poner a disposición del cliente el uso de la infraestructura informática (capacidad de computación, espacio de disco y bases de datos entre otros) como un servicio. Los clientes que optan por este tipo de familia cloud en vez de adquirir o

[23] URUEÑA E. V. Alberto, FERRARI Annie, BLANCO David. Op. cit.,
[24] MELL P. and GRANCE T.. Op. cit.,
[25] URUEÑA E. V. Alberto, FERRARI Annie, BLANCO David. Op. cit.
[26] Ibíd. Pág.

dotarse directamente de recursos como pueden ser los servidores, el espacio del centro de datos o los equipos de red, optan por la externalización en busca de un ahorro en la inversión en sistemas TI.

Con esta externalización, las facturas asociadas a este tipo de servicios se calculan en base a la cantidad de recursos consumidos por el cliente, basándose así en el modelo de pago por uso.

- **Software as a Service (SaaS):** Familia de cloud computing que consiste en la entrega de aplicaciones como servicio, siendo un modelo de despliegue de software mediante el cual el proveedor ofrece licencias de su aplicación a los clientes para su uso como un servicio bajo demanda.

  Los proveedores de los servicios SaaS pueden tener instalada la aplicación en sus propios servidores web (permitiendo a los clientes acceder, por ejemplo, mediante un navegador web), o descargar el software en los sistemas del contratante del servicio. En este último caso, se produciría la desactivación de la aplicación una vez finalice el servicio o expire el contrato de licencia de uso.

- **Platform as a Service (PaaS):** Familia de cloud computing que consiste en la entrega como un servicio de un conjunto de plataformas informáticas orientadas al desarrollo, testeo, despliegue, hosting y mantenimiento de los sistemas operativos y aplicaciones propias del cliente. Las principales características asociadas al Platform as a Service como solución cloud se exponen a continuación:

- Facilita el despliegue de las aplicaciones del cliente, sin el coste y la complejidad derivados de la compra y gestión del hardware y de las capas de software asociadas.
- Ofrece a través de redes de servicio IP todos los requisitos necesarios para crear y entregar servicios y aplicaciones web.

- **Business Process as a Service (BPaaS):** Familia de cloud computing que consiste en la provisión como servicio de procesos de negocio end-to-end altamente estandarizados a través de su entrega dinámica, la modalidad de pago por uso y los modelos de consumo de autoservicio bajo demanda. Su característica principal es que los recursos utilizados mediante esta solución para ejecutar los procesos de negocio, son compartidos entre los diferentes clientes del proveedor. En muchos casos, este hecho proporciona un aporte de valor al negocio; sin embargo, la solución BPaaS se encuentra fase incipiente, siendo todavía un modelo de negocio en el que los proveedores tan solo operan en la actualidad en nichos concretos.

Figura 15. Arquitectura de la nube

Fuente: NIST.gov

3.2.2 Beneficios de la computación en la nube: Debido a que la mayoría del software funciona por sí sola es una infraestructura muy rápida, permite actualizaciones constantemente, adaptable a grandes cambios, permite el acceso a la mayoría de los sistemas en cualquier lugar donde se encuentre y al no requerir tanto hardware es económica. La computación en la nube ofrece una cantidad de beneficios técnicos y económicos[27].

De acuerdo a encuestas[28] realizadas por la Agencia Europea de Seguridad de las Redes y de la Información (ENISA) a entidades localizadas en la Unión Europa, América y Asia entre las cuales se encuentran pymes españolas, las principales

[27] AYMERICH F. M., FENU G., and SURCIS S., "An approach to a cloud computing network," *1st Int. Conf. Appl. Digit. Inf. Web Technol. ICADIWT 2008*, pp. 113–118, 2008.
[28] An SME perspective on Cloud Computing, Surveyll. ENISA (European Network and Information Security Agency), 2009 [en linea] disponible en: http://www.enisa.europa.eu/activities/riskmanagement/files/deliverables/cloud-computing-sme-survey

ventajas que se aprecian e influyen a la hora de adoptar este tipo de soluciones en pequeñas y medianas empresas son el ahorro de costes de capital (68,1%) y la facilidad de aumentar los recursos disponibles (63,9%)[14].

Tabla 3. Motivos para la implementación de soluciones *cloud computing* (2009)

%	Razones
30,6%	Eliminación, mediante la incorporación de tecnologías de la información, de las barreras económicas y de conocimiento que impiden la modernización de los procesos de negocio.
68,1%	Evitar los gastos de capital en hardware, software, soporte de TI y seguridad de la información mediante la externalización de infraestructura/plataformas/servicios.
63,9%	Flexibilidad y escalabilidad de los recursos de TI.
36,1%	Aumento de la capacidad informática y del rendimiento del negocio.
11,1%	Diversificación de los sistemas de TI.
25%	Optimización local y global de la infraestructura de TI mediante la gestión automática de máquinas virtuales.
52,8%	Continuidad de negocio y capacidad de recuperación de desastres.
29,2%	Evaluación de la viabilidad y rentabilidad de nuevos servicios (como por ejemplo mediante el desarrollo de casos prácticos en la nube).
27,8%	Incorporar recursos redundantes para aumentar la disponibilidad y elasticidad de los mismos.
15,3%	Controlar los costes y beneficios marginales.
13,9%	Otros.

Fuente: An SME perspective on Cloud Computing, Surveyll. ENISA (European Network and Information Security Agency), 2009

3.2.2.1 Ventajas técnicas

- Maneja fácilmente situaciones de carga máxima sin necesidad de hardware adicional, los recursos pueden ser virtualizados y presentados a los clientes como los servidores virtuales que se gestionan a sí mismos. Físicamente, el recurso podría abarcar varios ordenadores o incluso múltiples centros de datos.

- El almacenamiento de datos en la nube tiene algunas ventajas sobre el acceso basado en el cliente. Es posible utilizar la potencia de procesamiento de la nube para hacer cosas que las aplicaciones tradicionales de productividad no pueden hacer.
- Permite utilizar los ordenadores del servidor y de las materias primas tradicionales, que además de ser más económico en términos de infraestructuras, son fáciles de mantener y recuperar. Por otra parte, a la luz de lo anterior, el sistema ofrece una mayor escalabilidad, ya que es posible añadir recursos computacionales por demanda del usuario, a veces también derivados de los servidores empresariales infrautilizados.

3.2.2.2 Ventaja para los usuarios

- Una de las mayores ventajas es que el usuario ya no está atado a una computadora tradicional al usar una aplicación, no tiene que comprar una versión configurada específicamente para un teléfono, PDA o cualquier otro dispositivo. En el futuro inmediato cualquier dispositivo que pueda acceder a Internet será capaz de ejecutar una aplicación basada en la nube.
- Independientemente del dispositivo que se utilice, puede haber menos problemas de mantenimiento. Los usuarios no tendrán que preocuparse por la capacidad de almacenamiento, compatibilidad u otros asuntos. Además, al tener acceso a una aplicación en la nube, el usuario puede estar seguro de obtener la última versión (sin tener que actualizar la versión almacenada disco duro).

3.2.2.3 Ventajas de la arquitectura

- La infraestructura de computación en la nube permite a las empresas lograr un uso más eficiente de sus inversiones en hardware y software de TI: aumenta la rentabilidad mediante la mejora de la utilización de recursos. Compartir los recursos en grandes nubes reduce los costes y aumenta la utilización mediante la entrega de recursos solamente por el tiempo que se necesitan usar.
- La Infraestructura de computación en la nube acelera y promueve la adopción de innovación en las empresas ya que estas se dan cuenta de que tienen que buscar nuevas ideas y abrir nuevas fuentes de valor, impulsadas por la presión para reducir costos y aumentar su crecimiento. La computación en nube alivia la necesidad de innovadores para encontrar recursos y así desarrollar, probar y hacer que sus innovaciones estén a disposición de la comunidad de usuarios.

La nube permite incluso al negocio más pequeño operar en la Web como una gran empresa[29].

El monitoreo y rastreo de las actividades en línea se han convertido en actividades más frecuentes para las organizaciones, la gente a menudo debe comprometer su privacidad para obtener acceso y por conveniencia. La mayoría de las empresas y marcas tienen como 'máxima prioridad' capturar y retener el interés de las personas con los datos acumulados sobre ellas. Esta tendencia se acelerará dado una mayor conectividad de la nube. Algoritmos automatizados analizan las interacciones sociales en línea, sobre todo cuando se utiliza un único proveedor

[29] "Servicios en la Nube para Pymes y sus Beneficios | Creativa Web®." [Online]. Available: http://www.creativaweb.co/servicios-en-la-nube-para-pymes/. [Accessed: 13-Sep-2015].

de la nube, como Amazon o Google. Cada vez que usted interactúa con un dispositivo, divulga detalles íntimos de su vida[30].

"De la misma manera que
vida inteligente evolucionó
para adaptarse a un cambio
de ecosistema, la nube
evolucionará a medida que nuestras necesidades
y las complejidades de
nuestros problemas evolucionan" ***Thomas M. Koulopoulos***

3.3 SERVICIOS WEB

El World Wide Web consortium (W3C)[*] describe los Servicios Web como, un conjunto de aplicaciones o de tecnologías con capacidad para interoperar en la Web y que intercambian datos entre sí con el objetivo de ofrecer servicios. Los proveedores ofrecen sus servicios como procedimientos remotos y los usuarios solicitan un servicio llamando a estos procedimientos a través de la Web. Estos servicios proporcionan mecanismos de comunicación estándar entre diferentes aplicaciones, que interactúan entre sí para presentar información dinámica al usuario.

Desde finales de 1990, ha surgido la computación orientada a servicios (SOC) como un importante paradigma y cambiando la forma en que aplicaciones de software están diseñadas, entregadas y consumidas, aunque la estructura de

[30] KOULOPOULOS T. M., *Cloud Surfing: A New Way to Think About Risk, Innovation, Scale and Success*. 2012.
[*] El World Wide Web Consortium (W3C) es una comunidad internacional que desarrolla estándares que aseguran el crecimiento de la Web a largo plazo.

servicios web ha sido fuertemente investigada, aún falta mucho por aprender, sobre todo dando la reciente subida de varios nuevos paradigmas de computación como la computación en nube, las redes sociales y el Internet de las Cosas[31].
En los últimos años se ha popularizado un estilo de arquitectura Software conocido como REST (Representational State Transfer). Este nuevo estilo ha supuesto una nueva opción de uso de los Servicios Web[32].

3.3.1 Servicios Web tipo REST

3.3.1.1 ¿Qué es REST realmente?: REST (Representational State Transfer) es una arquitectura de software para sistemas hipermedias distribuidos. REST se refiere estrictamente a una colección de principios para el diseño de arquitecturas en red[33]. El término se utiliza en el sentido de describir a cualquier interfaz que transmite datos específicos de un dominio sobre HTTP sin una capa adicional.

REST no es un estándar, es un estilo de arquitectura, se basa en los estándares:

- HTTP
- URL
- Representación de los recursos: XML/HTML/GIF/JPEG/…
- Tipos MIME: text/xml, text/html …

3.3.1.2 ¿Cuál es la motivación de REST?: Capturar las características de la Web que lo han hecho tan exitosa. La Web es la única aplicación distribuida que ha conseguido ser escalable al tamaño de Internet, el éxito lo debe al uso de formatos

[31] SHENG Q. Z., QIAO X., VASILAKOS A. V., SZABO C., BOURNE S., and XU X., "Web services composition: A decade's overview," *Inf. Sci. (Ny).*, vol. 280, pp. 218–238, 2014.
[32] MARSET R. N., *RESTRESTvsvsWeb ServicesWeb Services*. 2007.
[33] Ibíd. Pág.

de mensaje extensibles y estándares, pero además cabe destacar que posee un esquema de direccionamiento global (estándar y extensible a su vez).

3.3.1.3 ¿Cuáles son los principios de REST?: El estilo de arquitectura subyacente a la Web es el modelo REST, los objetivos de este estilo de arquitectura son:

- Escalabilidad de la interacción con los componentes. La Web no ha crecido exponencialmente sin degradar su rendimiento.
- Generalidad de interfaces. Gracias al protocolo HTTP, cualquier cliente puede interactuar con cualquier servidor HTTP sin ninguna configuración especial.
- Puesta en funcionamiento independiente. Los clientes y servidores pueden ser puestas en funcionamiento durante años. Por tanto, los servidores antiguos deben ser capaces de entender con clientes actuales y viceversa[34]. HTTP permite la extensibilidad mediante el uso de las cabeceras, a través de las URIs, a través de la habilidad para crear nuevos métodos y tipos de contenido.
- Compatibilidad con componentes intermedios. Lo más populares intermediarios son varios tipos de proxys para Web. Algunos de ellos, las caches, se utilizan para mejorar el rendimiento. Otros permiten reforzar las políticas de seguridad: firewalls. Y por último, otro tipo importante de intermediarios, Gateway, permiten encapsular sistemas no propiamente Web. Por tanto, la compatibilidad con intermediarios nos permiten reducir la latencia de interacción, reforzar la seguridad y encapsular otros sistemas.

[34] Ibíd. Pág.

REST logra satisfacer estos objetivos aplicando cuatro restricciones

- Identificación de recursos y manipulación de ellos a través de representaciones. Esto se consigue mediante el uso de URIs. HTTP es un protocolo centrado en URIs. Los recursos son los objetos lógicos a los que se le envían mensajes. Los recursos no pueden ser directamente accedidos o modificados. Más bien se trabaja con representaciones de ellos.
- Mensajes auto descriptivos. REST dicta que los mensajes HTTP deberían ser tan descriptivos como sea posible. Esto hace posible que los intermediarios interpreten los mensajes y ejecuten servicios en nombre del usuario. Uno de los modos que HTTP logra esto es por medio del uso de varios métodos estándares, muchos encabezamientos y un mecanismo de direccionamiento. HTTP es un protocolo sin estado y cuando se utiliza adecuadamente, es posible interpretar cada mensaje sin ningún conocimiento de los mensajes precedentes.
- Hipermedia como un mecanismo del estado de la aplicación. El estado actual de una aplicación Web debería ser capturada en uno o más documentos de hipertexto, residiendo tanto en el cliente como en el servidor. El servidor conoce sobre el estado de sus recursos, el navegador sabe cómo navegar de recurso a recurso, recogiendo información que el necesita o cambiar el estado que el necesita cambiar.

Aquellos que han decidido adoptar REST como un modelo de servicios Web sienten que al menos articula una filosofía de diseño con fortaleza, debilidades y áreas de aplicabilidad documentada.

3.3.1.4 Características de REST:

Tabla 4. Características, Ventajas y Desventajas de REST

	REST
Características	Las operaciones se definen en los mensajes. Una dirección única para cada instancia del proceso. Cada objeto soporta las operaciones estándares definidas. Componentes débilmente acoplados.
Ventajas declaradas	Bajo consumo de recursos Las instancias del proceso son creadas explícitamente. El cliente no necesita información de enrutamiento a partir de la UTI inicial. Los clientes pueden tener una interfaz "listener" (escuchadora) genérica para las notificaciones. Generalmente fácil de construir y adoptar.
Posibles desventajas	Gran número de objetos Manejar el espacio de nombres (URIs) puede ser engorroso. La descripción sintáctica/semántica muy informal (orientada al usuario). Pocas herramientas de desarrollo.

Fuente: (Marset, 2007)

3.3.1.5 ¿Qué pasará con REST?: Los negocios electrónicos van a necesitar algo más que tecnologías orientadas en RCP. Todos los negocios de cualquier lugar tendrán que estandarizar sus modelos de direccionamiento para exponer las interfaces en común a sus socios. Para que los negocios interoperen sin programar manualmente de manera explícita enlaces a los socios, se necesitará estandarizar un modelo de direccionamiento, más que invertir en sistemas propietarios. REST proporciona un alto grado de estandarización[35].

[35] Ibíd. Pág.

3.4 MODELO VISTA CONTROLADOR (MVC)

Hoy día en cualquier lugar del mundo los que construyen aplicaciones informáticas centran su atención en dos aspectos fundamentales: cómo lograr construir mejores aplicaciones en menos tiempo, y cómo utilizar mayor cantidad de estándares en el diseño de las aplicaciones que permitan mayor reutilización del código y mejores mantenimientos de los sistemas desarrollados[36].

El Modelo-Vista-Controlador (MVC) es considerado como un patrón de diseño, surge con el objetivo de reducir el esfuerzo de programación, necesario en la implementación de sistemas múltiples y sincronizados de los mismos datos, a partir de estandarizar el diseño de las aplicaciones. El patrón MVC es un paradigma que divide las partes que conforman una aplicación en el Modelo, las Vistas y los Controladores, permitiendo la implementación por separado de cada elemento, garantizando así la actualización y mantenimiento del software de forma sencilla y en un reducido espacio de tiempo. A partir del uso de frameworks basados en el patrón MVC se puede lograr una mejor organización del trabajo y mayor especialización de los desarrolladores y diseñadores.

3.4.1 Historia: El patrón MVC fue una de las primeras ideas en el campo de las interfaces gráficas de usuario y uno de los primeros trabajos en describir e implementar aplicaciones software en términos de sus diferentes funciones.

A principios de los años setenta, Alan Kay fundó el *Learning Research Group* en el *Palo Alto Research Center* (PARC) de Xerox. Kay investigó cómo hacer a los

[36] P. Modelo-vista-controlador, “Patrón Modelo-Vista-Controlador.,” vol. 11, no. 1, pp. 47–57, 2012.

ordenadores más potentes y fáciles de usar, teniendo siempre en mente el Dynabook, su visión de lo que debía ser un ordenador personal[37].

Fue en este centro donde se introdujo el termino MVC por Trygve Reenskaug en Smalltalk*-76 durante su visita a Xerox Parc[38] en los años 70 y, seguidamente, en los años 80, Jim Althoff y otros implementaron una versión de MVC para la biblioteca de clases de Smalltalk-80, Aunque la intención era apoyar interfaces gráficas para aplicaciones desarrolladas en Smalltalk, las herramientas del entorno de Smalltalk como los navegadores, en particular también fueron implementados utilizando el paradigma MVC.Solo fue hasta 1988, cuando MVC se expresó como un concepto general en un artículo[39] sobre Smalltalk-80.

3.4.2 Descripción del patrón: Se puede decir según J. S. Castejón Garrido[40], que este patrón está conformado por tres capas:

- *Nivel de presentación:* es el encargado de generar la interfaz de usuario en función de las acciones llevadas a cabo por el mismo.
- *Nivel de negocio:* contiene toda la lógica que modela los procesos de negocio y es donde se realiza todo el procesamiento necesario para atender a las peticiones del usuario.
- *Nivel de administración de datos*: encargado de hacer persistente toda la información, suministra y almacena información para el nivel de negocio.

[37] "Capítulo 6: Alan Kay y la metáfora del escritorio." [Online]. Available: http://www.adelal.com/nOproblemO/LVR/Cap6.htm. [Accessed: 12-Sep-2015].
[*] Smalltalk: lenguaje de propagación
[38] "Trygve/MVC." [Online]. Available: http://heim.ifi.uio.no/~trygver/themes/mvc/mvc-index.html. [Accessed: 12-Sep-2015].
[39] KRASNER G. E. and POPE S. T., "A cookbook for using the model-view controller user interface paradigm in Smalltalk-80," *J. Object-Oriented Program.*, vol. 1, no. 3, pp. 26–49, Aug. 1988.
[40] CASTEJÓN GARRIDO J. S., "Arquitectura y diseño de sistemas web modernos," *Rev. Ing. Informáctica del CIIRM*, pp. 1–6, 2004.

De manera genérica, estas tres capas son más conocidas si se definen así,

- **El Modelo:**

  Es la representación de la información con la cual el sistema opera, por lo tanto gestiona todos los accesos a dicha información, tanto consultas como actualizaciones, implementando también los privilegios de acceso que se hayan descrito en las especificaciones de la aplicación (lógica de negocio). Envía a la 'vista' aquella parte de la información que en cada momento se le solicita para que sea mostrada (típicamente a un usuario). Las peticiones de acceso o manipulación de información llegan al 'modelo' a través del 'controlador'.

- **El Controlador:**

  Responde a eventos (usualmente acciones del usuario) e invoca peticiones al 'modelo' cuando se hace alguna solicitud sobre la información (por ejemplo, editar un documento o un registro en una base de datos). También puede enviar comandos a su 'vista' asociada si se solicita un cambio en la forma en que se presenta de 'modelo' (por ejemplo, desplazamiento o scroll por un documento o por los diferentes registros de una base de datos), por tanto se podría decir que el 'controlador' hace de intermediario entre la 'vista' y el 'modelo' (véase Middleware).

- **La Vista:**

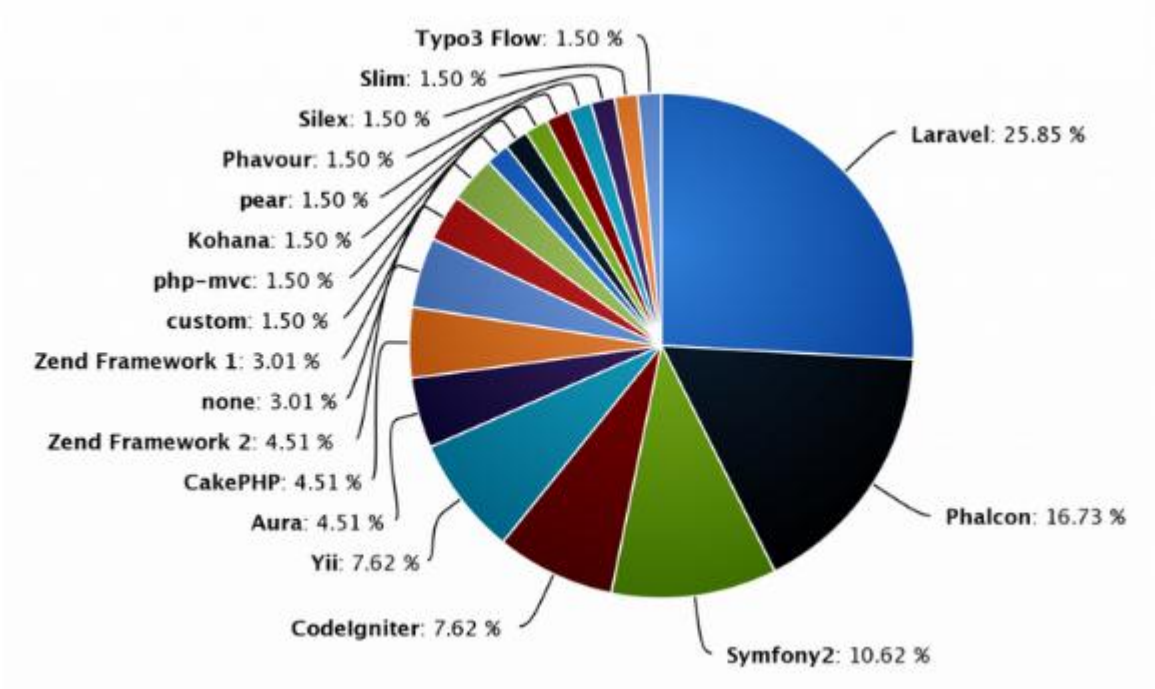
  Presenta el 'modelo' (información y lógica de negocio) en un formato adecuado para interactuar (usualmente la interfaz de usuario) por tanto requiere de dicho 'modelo' la información que debe representar como salida.

3.5 HERRAMIENTAS UTILIZADAS

3.5.1 Frameworks: Para el desarrollo de aplicaciones informáticas es importante contar con la normalización de sus datos, para que la información este estructurada y sea más fácil manejarla, almacenarla y hasta recuperarla.

Existe una gran cantidad de herramientas que pueden servir cuando se quiere desarrollar un proyecto basado en el patrón de diseño MVC, en la *Figura 16* se pueden observar los mejores frameworks* existentes hasta el año 2014.

Figura 16. Principales Frameworks php 2014



Fuente: SITEPOINT best php frameworks [en linea] disponible en: http://www.sitepoint.com/best-php-frameworks-2014/

* Framework: Es un esquema, patrón o marco de trabajo, que define unos estándares para el desarrollo y/o implementación de una aplicación software.

3.5.1.1 Framework Yii: Según el sitio oficial[41], Yii es un framework PHP de alto rendimiento basado en componentes para el desarrollo de aplicaciones web a gran escala. Permite la máxima reutilización en la programación Web y puede acelerar significativamente el proceso de desarrollo de aplicaciones Web.

El nombre Yii (pronunciado Yee o [ji:]) es un acrónimo para "Sí lo es". Esta suele ser la correcta y la respuesta más concisa a las consultas de los nuevos en Yii: ¿Es rápido? ... ¿Es seguro? ... ¿Es profesional? ... ¿Es correcto para mi próximo proyecto? ... ¡Sí lo es!.

Yii es un framework de programación Web genérica que se puede utilizar para el desarrollo de prácticamente cualquier tipo de aplicación web. Debido a que es ligero y equipado con sofisticados mecanismos de caché, es especialmente adecuado para aplicaciones de alto tráfico, tales como portales, foros, sistemas de gestión de contenidos (CMS), sistemas de comercio electrónico, etc. Como la mayoría de los frameworks de PHP, Yii es un framework MVC.

Yii sobresale entre los frameworks de PHP en ser eficiente, rico en características y claramente documentado. Yii ha sido cuidadosamente diseñado desde cero con el fin de desarrollar aplicaciones web serias. No es ni un subproducto de algún proyecto ni un conglomerado de trabajo de terceros. Es el resultado de la rica experiencia de los autores con el desarrollo de aplicaciones Web y su investigación de los frameworks de programación web y las aplicaciones más populares

[41] YIIFRAMEWORK quickstart what is yii [en linea] disponible en: http://www.yiiframework.com/doc/guide/1.1/en/quickstart.what-is-yii

3.5.2 Sublime Text: El sitio oficial de sublime text[42] lo define como un editor sofisticado de código multiplataforma, el cual soporta un gran número de lenguajes (C, C++, C#, CSS, D, Erlang, HTML, Groovy, Haskell, HTML, Java, JavaScript, LaTeX, Lisp, Lua, Markdown, Matlab, OCaml, Perl, PHP, Python, R, Ruby, SQL, TCL, Textile and XML). Sublime Text está disponible para OS X, Windows y Linux.

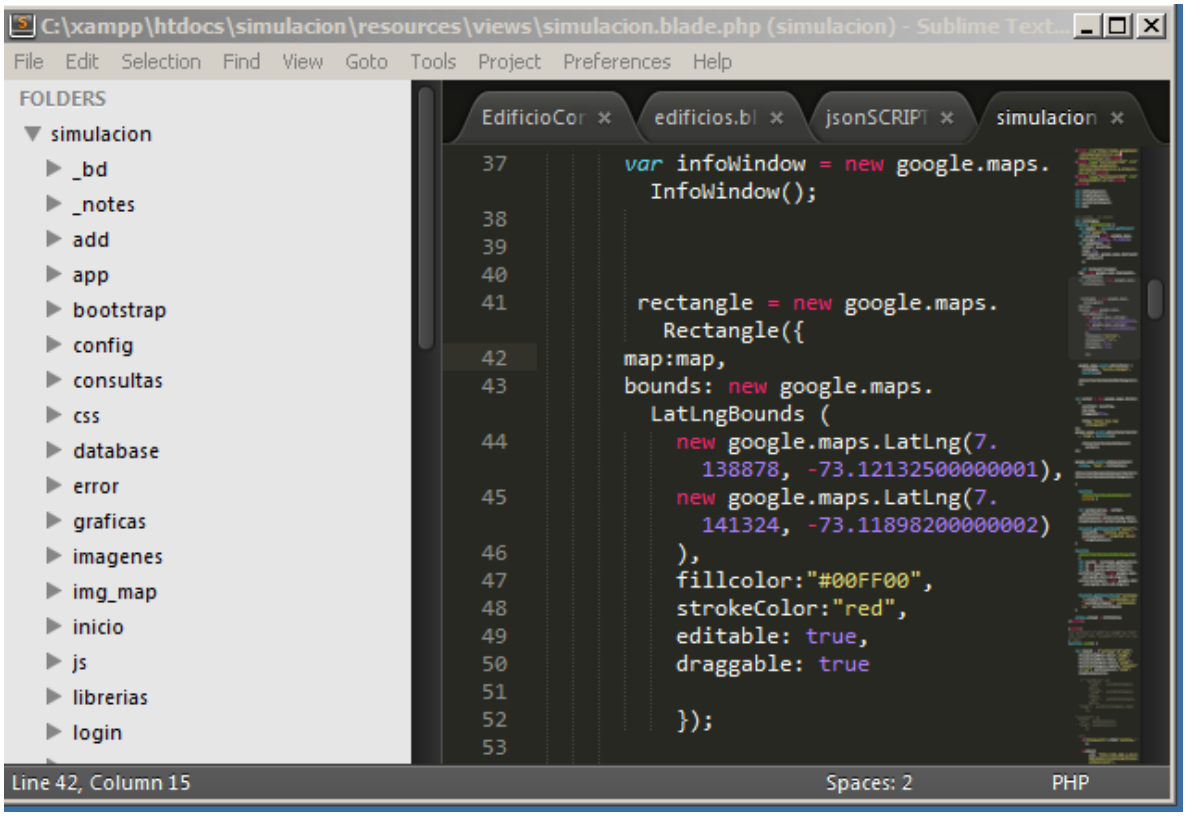
El programa dispone de auto-guardado, muchas opciones de personalización, cuenta con un buen número de herramientas para la edición del código y automatización de tareas. Soporta macros, *Snippets* y auto completar, también permite hacer cambios de forma interactiva en muchas líneas a la vez, cambiar el nombre de las variables con facilidad, y manipular los archivos más rápido, entre otras funcionalidades. Algunas de sus características son ampliables mediante plugins.

Sublime Text ver *Figura 17,* permite tener varios documentos abiertos mediante pestañas, e incluso emplear varios paneles para aquellos que utilicen más de un monitor. Dispone de modo de pantalla completa, para aprovechar al máximo el espacio visual disponible de la pantalla.

El programa cuenta con 22 combinaciones de color posibles, aunque se pueden conseguir más. Para navegar por el código cuenta con *Minimap*, un panel que permite moverse por el código de forma rápida.

[42] http://www.sublimetext.com/

Figura 17. Sublime Text De Jong



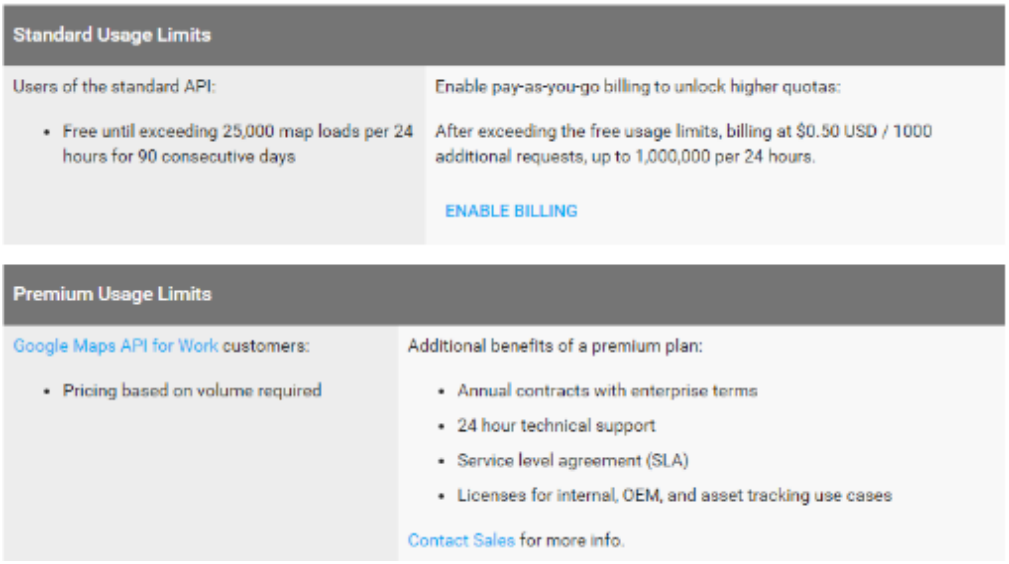
3.5.3 Google Maps: Es un servidor de aplicaciones de mapas en la web que pertenece a Google. Ofrece imágenes de mapas desplazables, así como fotografías por satélite del mundo e incluso la ruta entre diferentes ubicaciones. Google Maps fue desarrollado originalmente por dos hermanos Daneses, Lars y Jens Rasmussen, co-fundadores de Where 2 Technologies una empresa dedicada a la creación de soluciones de mapeo. La empresa fue adquirida por Google en octubre de 2004, y los dos hermanos luego crearon Google Maps.

El API de Google Maps, consiste de archivos JavaScript que contienen las clases, métodos y propiedades que se usan para el comportamiento de los mapas, antes de que hubiera una API pública, algunos desarrolladores descubrieron la manera de hackear Google Maps para incorporar los mapas en sus propios sitios web.

Esto llevó a Google a la conclusión de que había la necesidad de una API pública, y en junio de 2005 fue lanzada públicamente. En mayo de 2010, se anunció la versión 3 del API. Ahora es la opción recomendada para las nuevas aplicaciones de Google Maps.

Para utilizar la API de Google Maps, en las versiones anteriores a la V3.0 se requería añadir una API_KEY al código, esta clave de uso tiene un límite gratuito de 25.000 solicitudes por día ver *Figura 18*, permite tener estadísticas de uso, un resumen del estado de facturación y si se desea, el usuario puede comprar una cuota más alta de solicitudes.

Figura 18. Restricciones API_KEY Google maps

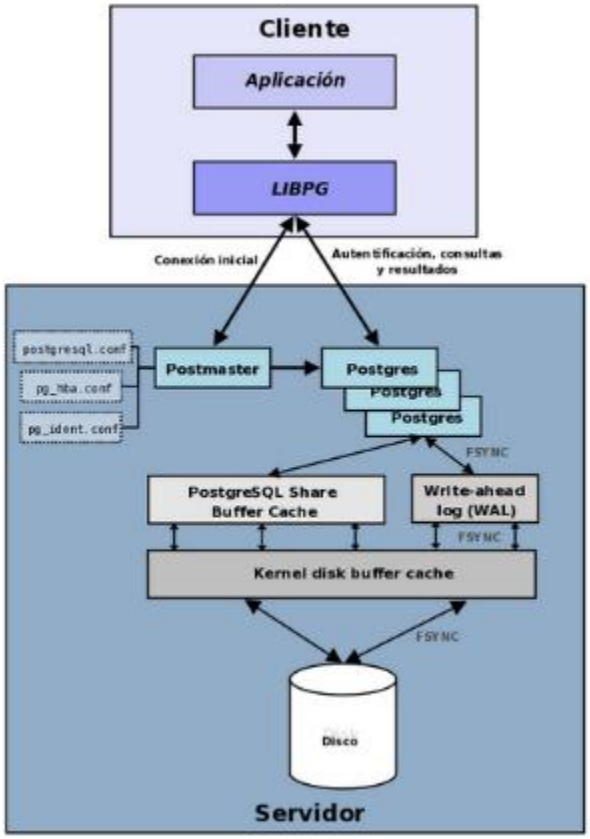


Fuente: DEVELOPERS [en línea] disponible en: https://developers.google.com/maps/documentation/*javascript/usage?hl=es*

3.5.4 PostgreSQL: PostgreSQL es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia BSD y con su código fuente disponible libremente. Es el sistema de gestión de bases de datos de código abierto más

avanzada y potente del mercado. PostgreSQL utiliza un modelo cliente/servidor y usa *multiprocesos* en vez de *multihilos* para garantizar la estabilidad del sistema por lo que un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando[43].

Figura 19. Componentes más importantes en un sistema PostgreSQL



Fuente: POSTGRESQL sobre postgresql [en línea] disponible en: http://www.postgresql.org.es/sobre_postgresql

Los componentes más importantes en un sistema PostgreSQL, se pueden observar en la *Figura 19* y se describen a continuación,

[43] Postgresql.org.es, "Sobre PostgreSQL." [Online]. Available: http://www.postgresql.org.es/sobre_postgresql. [Accessed: 06-Oct-2015].

- **Aplicación cliente**: Esta es la aplicación cliente que utiliza PostgreSQL como administrador de bases de datos. La conexión puede ocurrir vía TCP/IP o sockets locales.
- **Demonio postmaster**: Este es el proceso principal de PostgreSQL; Es el encargado de escuchar por un puerto/socket las conexiones entrantes de clientes, también es el encargado de crear los procesos hijos que se encargaran de autentificar estas peticiones, gestionar las consultas y mandar los resultados a las aplicaciones clientes.
- **Ficheros de configuración**: Los 3 ficheros principales de configuración utilizados por PostgreSQL, son postgresql.conf, pg_hba.conf y pg_ident.conf.
- **Procesos hijos postgres**: Los procesos hijos que se encargan de autentificar a los clientes, de gestionar las consultas y mandar los resultados a las aplicaciones clientes.
- **PostgreSQL share buffer cache**: Memoria compartida usada por POstgreSQL para almacenar datos en caché.
- **Write-Ahead Log (WAL)**: Componente del sistema encargado de asegurar la integridad de los datos (recuperación de tipo REDO)
- **Kernel disk buffer cache**: Caché de disco del sistema operativo
- **Disco**: Disco físico donde se almacenan los datos y toda la información necesaria para que PostgreSQL funcione.

3.5.4.1 Altamente personalizable: PostgreSQL ejecuta procedimientos almacenados en más de una docena de lenguajes de programación, incluyendo Java, Perl, Python, Ruby, Tcl, C / C ++, y su propio PL / pgSQL, que es similar a la de Oracle PL / SQL. Incluye en su biblioteca de funciones, cientos de funciones estándar integradas que van desde las operaciones básicas de matemáticas y cadenas de operaciones para la criptografía y la compatibilidad con Oracle. Privilegios y procedimientos almacenados pueden ser escritos en C y se cargan

en la base de datos como una biblioteca, lo que permite una gran flexibilidad en la ampliación de sus capacidades.

Del mismo modo, PostgreSQL incluye un marco que permite a los desarrolladores definir y crear sus propios tipos de datos personalizados junto con funciones de apoyo y operadores que definen su comportamiento.

El Proyecto PostgreSQL tiene como objetivo mantener y soportar cada versión de PostgreSQL durante 5 años desde el momento de su lanzamiento.

3.5.4.2 Ventajas: Las ventajas descritas a continuación son una recopilación de información del sitio oficial de postgreSQL[44].

1. Instalación Ilimitada Con PostgreSQL, nadie puede demandarlo por violar acuerdos de licencia, puesto que no hay costo asociado a la licencia del software.
2. Soporte Además de nuestras ofertas de soporte, tenemos una importante comunidad de profesionales y entusiastas de PostgreSQL de los que su compañía puede obtener beneficios y contribuir.
3. Ahorros considerables en costos de operación PostgreSQL ha sido diseñado y creado para tener un mantenimiento y ajuste mucho menor que otros productos, conservando todas las características, estabilidad y rendimiento.
4. Estabilidad y Confiabilidad Legendarias Es extremadamente común que compañías reporten que PostgreSQL nunca ha presentado caídas en varios años de operación de alta actividad. Ni una sola vez. Simplemente funciona.
5. Extensible El código fuente está disponible para todos sin costo. Si su equipo necesita extender o personalizar PostgreSQL de alguna manera, pueden hacerlo con un mínimo esfuerzo, sin costos adicionales. Esto es

[44] "PostgreSQL: Advantages." [Online]. Available: http://www.postgresql.org/about/advantages/. [Accessed: 06-Oct-2015].

complementado por la comunidad de profesionales y entusiastas de PostgreSQL alrededor del mundo que también extienden PostgreSQL todos los días.

6. Multiplataforma PostgreSQL está disponible en casi cualquier Unix (34 plataformas en la última versión estable), y ahora en versión nativa para Windows.
7. Diseñado para ambientes de alto volumen PostgreSQL usa una estrategia de almacenamiento de filas llamada MVCC para conseguir una mejor respuesta en ambientes de grandes volúmenes. Los principales proveedores de sistemas de bases de datos comerciales usan también esta tecnología, por las mismas razones.
8. Herramientas gráficas de diseño y administración de BD Existen varias herramientas gráficas de alta calidad para administrar las bases de datos (pgAdmin , pgAccess) y para hacer diseño de bases de datos (Tora , Data Architect).

Para el proyecto la herramienta de administración que se implemento fue PgAdmin3, Es una interfaz comprensible para el diseño y administración de una base de datos PostgreSQL, diseñada para ejecutarse en la mayoría de los Sistemas Operativos. La aplicación corre bajo GNU/Linux, FreeBSD y Windows 2000/XP. La interfaz gráfica soporta todas las características de PostgreSQL y facilita la administración.

4. SENSORES VIRTUALES

Sensores virtuales o también llamados "soft sensors", son algoritmos que usan mediciones provenientes de sensores reales o datos de otras fuentes, que pueden ser usados en la predicción del comportamiento de situaciones que requieren planeación o un nivel considerable de prevención en su ejecución. El término sensor virtual es nuevo, se puede definir como la asociación de sensores tradicionales (hardware) con un algoritmo de estimación (software), con el fin de proporcionar estimaciones en línea de variables no medidas, de variables con tiempos muertos en la medición, o de parámetros de un modelo[45].

Actualmente los sensores virtuales se usan para el monitoreo de líneas de oleoductos y la simulación del comportamiento de pacientes médicos con el fin de anticipar el comportamiento de una enfermedad[46].

4.1 PLATAFORMA DE SERVICIOS BASADA EN SENSORES VIRTUALES

Las simulaciones constituyen un complemento perfecto para la telemetría y consecuentemente para apoyar la gestión del espectro radioeléctrico. Surge entonces la necesidad de integrar las simulaciones a las plataformas de servicios en las que convergen la electrónica y la informática. Para lograrlo resulta necesario revisar previamente la arquitectura más conveniente para este tipo de

[45] J. C. B. Gonzaga, L. a C. Meleiro, C. Kiang, and R. Maciel Filho, "ANN-based soft-sensor for real-time process monitoring and control of an industrial polymerization process," *Comput. Chem. Eng.*, vol. 33, pp. 43–49, 2009.
[46] B. M. Lee and U. G. Kang, "Virtual Sensor for Diabetes Meter in U-Health Service," vol. 6, no. 6, pp. 87–96, 2014.

implementaciones teniendo en cuenta las apuestas nacionales e internacionales y las ventajas que puede tener para impulsar el desarrollo de servicios en los que convergen las más diversas profesiones y tecnologías.

En una arquitectura IIoT un nuevo modelo de desarrollo de aplicaciones que la empresa National Instruments denomina el Internet de las Cosas Industriales[47] los equipos de medición pueden ser tratados como sensores que proporcionan información georeferenciada a un sistema de procesamiento en la nube. Precisamente los algoritmos de radiopropagación proporcionan también resultados, que aunque son simulados, corresponden a mediciones georeferenciadas. Es aquí cuando surge la necesidad de implementar el algoritmo De Jong como un sensor virtual en una plataforma IoT de apoyo a la gestión del ERE.

4.2 PLATAFORMA DE SENSORES VIRTUALES

En la plataforma propuesta para este trabajo mostrada en la *Figura 20,* se quiere demostrar cómo pueden o deben convivir los sensores virtuales con los reales ya que a futuro ellos deberán complementarse. Para esta plataforma se utilizó el concepto del patrón de diseño MVC.

Capa de datos: En esta capa se encuentran los datos de mediciones proporcionados por sensores reales o datos de simulaciones realizadas por sensores virtuales que ayudan a la lógica del negocio, esta capa hace parte del modelo de la aplicación. Contar con bases de datos es siempre necesario, pues

[47] T. Trends, T. Accelerate, and Y. Productivity. Op. cit.,

de otra manera las mediciones y los resultados que entregan los modelos serían volátiles.

Lógica de negocio: Es la capa encargada de la lógica y la algoritmia que simula el comportamiento de un sensor real. Lo interesante es que un programador puede ver los sensores virtuales de igual manera que los reales; esta es la razón por la cual en esta capa se han ubicado también los sensores reales.

Capa de servicios: En esta capa se configuran servicios que satisfacen necesidades concretas para los usuarios. Se trata básicamente de aplicaciones que pueden tener forma de web services o no, pero que atienden al usuario haciendo uso de los recursos disponibles en la lógica de negocio y la capa de datos, e incluso eligiendo las vistas que más convienen al usuario.

Vista: Es la capa donde acceden los usuarios finales, son las interfaces de la plataforma, pueden ser aplicaciones móviles, aplicaciones de terceros, drivers. Con esta visión un terminal de usuario podría ser tan simple como para soportar la vista, mientras el grueso de la aplicación corre en la nube.

Para demostrar la diferencia en el procesamiento de los datos de un sensor virtual comparado con un sensor real, se puede explicar con un ejemplo sencillo, se cuenta con un “drone” al que se le instala un sensor encargado de medir la radiación no ionizante (NIR). Desde el punto de vista de la plataforma de programación, el medidor ubicado en el drone es visto como un sensor en forma de un Web Service de medición real. El usuario cuenta con una aplicación “la vista” del servicio para ordenar el inicio y registro de las mediciones, esta orden es recibida por un servicio de mediciones considerado como el controlador y que procede a consumir el Web Service de medición real que finalmente deposita los

datos de medición en la base de datos correspondiente en la nube. Para la visualización, el usuario accede a una nueva vista de la interfaz de usuario que se comunica con un nuevo servicio o controlador alojado en la nube, este elige los web service que sean necesarios para presentar la información que necesita el usuario.

La descripción es similar usando un sensor virtual. Primero el usuario visualiza una vista a través de cualquier tipo de terminal, y solicita el servicio de simulación de sensores, luego este servicio se conecta con la lógica de negocio que tiene el algoritmo De Jong alojado en la nube, la cual se encarga de procesar y retornar una respuesta, si el usuario lo desea accede a una nueva vista de la interfaz de usuario que se comunica con un nuevo servicio o controlador alojado en la nube, este elige los web service que sean necesarios para presentar la información que necesita el usuario.

Con esta descripción se trata de explicar que para este tipo de plataforma propuesto los sensores reales tendrían el mismo trato que los sensores virtuales.

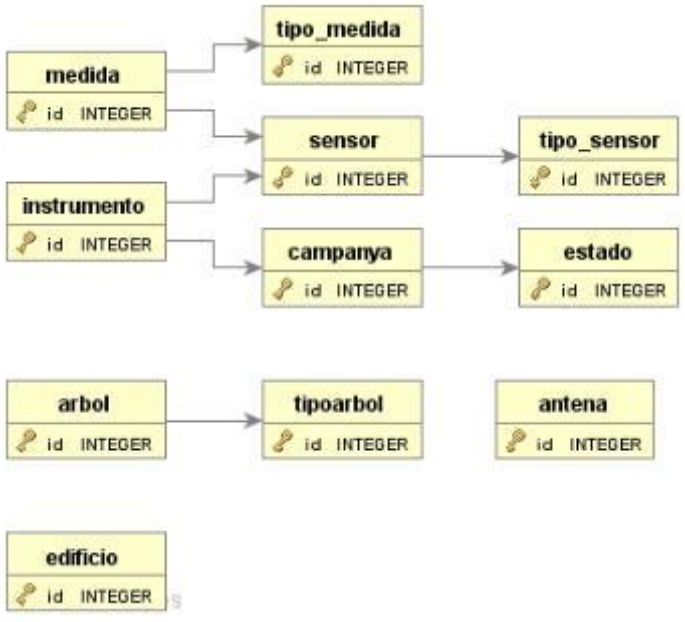
Figura 20. Plataforma MVC Sensores Virtuales y Reales

5. DESARROLLO

Una problemática que planteo el grupo de investigación RadioGis al iniciar este proyecto, era que los productos realizados en tesis anteriores de pregrado y maestría no estaban siendo utilizados para ningún fin, entonces surgió la necesidad de investigar algunos de estos productos para saber si se podían complementar con la solución de este proyecto de grado; después de investigar se encontró una plataforma web para la Monitorización del Espectro y la Radiación llamada “UISpectrum”, esta aplicación ofrecía servicios relacionados con la consulta web de mediciones del espectro ERE, la radiación no ionizante RNI, además se podía consultar la mancha de radiación existente para algunas ciudades proporcionada por mediciones reales tomadas por los investigadores; esta aplicación cumplía con un papel fundamental para la gestión del espectro, y también para evidenciar la propuesta planteada en esta tesis de grado, “los sensores virtuales como los reales tienen el mismo comportamiento” por estas razones se decidió complementar esta plataforma con un nuevo servicio, el servicio de simulación De Jong basado en la lógica de un sensor real.

Lo primero que se hizo fue desarrollar el modelo de datos *Figura 24*, para el modulo que va a complementar la plataforma UISpectrum. Desarrollar el modelo antes de iniciar el proyecto permite que todos los integrantes del equipo entiendan mejor el objetivo del proyecto y estén más claros sobre lo que se va a construir; en la figura se muestran las tablas que conforman la base de datos del algoritmo De Jong.

Figura 21. Modelo de datos Sensor virtual De Jong



Desde el principio del proyecto se instaló la aplicación opensource “IceSrum” mencionada en el capítulo metodología de este libro, esta herramienta fue utilizada para gestionar el proyecto bajo la metodología agile Scrum; Se crearon tres tipos de roles, el primero “product owner” para quienes fueron los encargados de trasladar la visión del proyecto al equipo, los directores del proyecto, el Profesor Homero ortega Boada de la E3T, y el profesor Manuel Guillermo Flórez de la escuela de Ingeniería de sistemas, el segundo rol fue el de “Scrum Master” que fue hecho para la persona que lideró al equipo guiándolo para que cumpliera las reglas y procesos de la metodología, el Ingeniero de Sistemas Egresado de la UIS Juan David Rodríguez Ariza, el tercer rol creado fue el “Team” para los miembros del grupo, los estudiantes María Alejandra Alfonso Muñoz y César Augusto Pico, encargados de desarrollar el proyecto de manera conjunta llevando a cabo las *historias* a las que se comprometen al inicio de cada sprint.

Al iniciar el proyecto se realizaron sesiones semanales entre todos los miembros del equipo, y a medida que se fueron teniendo claros los requisitos o *historias* por parte de los product owner se procedió a construir el *Product Backlog* el cual contiene los objetivos de este proyecto; se decidieron hacer 4 Sprint en el proyecto con una duración de 6 semanas cada uno, estos Sprint contienen la lista de tareas o también llamado el *Sprint Backlog* que se debía cumplir antes de acabar cada Ciclo.

Cumpliendo con la metodología, todos los días el equipo de desarrollo antes de iniciar a trabajar en las tareas del día, se reunía 15 minutos a explicar el tablero de tareas, véase la *Figura 21*, cada miembro mencionaba las tareas terminadas o que estaban en progreso, los inconvenientes que se presentaron y las soluciones que se encontraron. Al finalizar cada Sprint se presentaban las historias conseguidas mediante una demonstración a los product owner.

Figura 22. Tareas Iniciales IceScrum Sprint 1

En la *Figura 22* observamos una Lista de tareas o también llamado “Sprint Backlog” que el equipo estuvo elaborando para el segundo Sprint y así cumplir con los objetivos/requisitos definidos en el Product Backlog.

Esta lista permite ver las tareas donde el equipo está teniendo problemas y no avanza, con lo que le permite tomar decisiones al respecto. Para cada uno de los objetivos/requisitos se muestran sus tareas, el esfuerzo pendiente para finalizarlas.

Figura 23. Pila de iteración IceScrum

Pila de Iteración

Dejong

Lista de Tareas

ID	Asociado con	Nombre	Estado	Tiempo remanente	Responsable	Descripción	Notas	Creador
32	Tarea urgente	Buscar si hay una base de datos de arboles en la uis	Terminada	0.0	Maria Alejandra Alfonso Muñoz	Se debe consultar quienes fueron las personas encargadas de marqcar los arboles en la UIS, para preguntarles si nos podrian dar los registros tomados en digital o papel.		Maria Alejandra Alfonso Muñoz
39	Tarea urgente	maquina en la nube	Terminada	0.0	Cesar Pico	instalar lo necesario en la maquina virtual, para el proyecto		Cesar Pico
40	Tarea urgente	preguntar sobre BD de edificios	Terminada	0.0	Cesar Pico			Maria Alejandra Alfonso Muñoz
45	Tarea urgente	Usar laravel para consultar	Terminada	0.0	Maria Alejandra Alfonso Muñoz	Consultar datos de una tabla usando laravel		Juan David Rodriguez Ariza
14	Ingresar ROI	Montar ROI en mapa	Terminada	0.0	Maria Alejandra Alfonso Muñoz	Desde Google Maps Poder dibujar un cuadro con la zona de interes, la ubicacion del sensor, un boton de procesar y enviar al ws para q simule, debera responder el valor.		Juan David Rodriguez Ariza
38	Registro de usuarios	Menu de simulacion	Terminada	0.0	Maria Alejandra Alfonso Muñoz	Mostrar el mapa de dejong al dar click en el menu de UisSpectrum		Juan David Rodriguez Ariza
37	Registro de usuarios	Dar permisos al usuario	Terminada	0.0	Juan David Rodriguez Ariza	Dar permisos al usuario		Juan David Rodriguez Ariza
31	Registro de	Copia del	Terminada	0.0	Maria Alejandra	tener una copia del servidor de		Cesar Pico

Reporte de Scrum generado con iceScrum lunes 06 abril 2015 Página 1 / 5

Para desarrollar el servicio de simulación del algoritmo De Jong en una infraestructura web, primero se investigó sobre la implementación en Java del algoritmo desarrollado por estudiantes del grupo de investigación RadioGis de la

escuela de Ingenierías Eléctrica Electrónica y Telecomunicaciones de la Universidad Industrial de Santander UIS, para su tesis de pregrado[48]. El algoritmo basado en *ray-tracing*[*], se funda en la creación de varios emisores, denominados fuentes virtuales pues en su posición no existe un emisor físico sino uno creado por el algoritmo para simular los efectos de *reflexión, difracción* o *scattering,* estos fenómenos se describen anteriormente en el marco teórico.

El algoritmo se encontraba programado en el entorno de desarrollo Netbeans, debido a que utilizaba mucho espacio en memoria y el rendimiento era muy lento, se decidió migrar a Eclipse otro entorno de desarrollo más adaptado a las necesidades de nuestro caso.

Se crearon los mockups de la aplicación Anexo 1, estos son un diseño prototipo de la aplicación web, que permiten tener una visión más clara del alcance del proyecto, en la *Figura 23,* se puede observar a un usuario dentro de la aplicación, seleccionando la región de interés (RoI), agregando algunos árboles, edificios, y antenas.

[48] MEJÍA Y. and PINEDA M.. Op. cit.,

[*] Algoritmo que consiste en trazar los rayos para determinar las superficies visibles con un proceso de sombreado, que tiene en cuenta efectos globales de iluminación como pueden ser reflexiones, refracciones o sombras arrojadas.

Figura 24. Mockup Plataforma de servicios De Jong

Para el desarrollo de la plataforma se inició trabajando con Laravel pues es un framework PHP muy completo, y se ha convertido en el framework de PHP más popular, pero durante el tiempo que se utilizó se evidencio que muchas librerías que no se estaban utilizando más adelante se decidió migrar a Yii, porque se encontró que este framework era más eficiente y se adaptaba mejor al proyecto, tenía muchas ventajas con respecto a Laravel, una de las ventajas es que Yii tiene una curva de aprendizaje más pequeña por lo que tomó menos tiempo aprender su funcionamiento, además tiene bien claro y documentado el patrón de diseño MVC.

Luego de revisar cuidadosamente el funcionamiento del algoritmo desde el entorno de desarrollo eclipse, se empezó a migrar la aplicación de escritorio a un

servicio en la web, la primera tarea para llevar a cabo la migración a la web, fue convertir el algoritmo de un proyecto java a un proyecto java Maven el formato que permite realizar un web service, al realizar la conversión a Maven la programación hecha en JAVA queda intacta, simplemente hay que ser precavidos ya que es posible que se tenga que agregar algunas instancias para que el web service funcione con las librerías que usa el algoritmo original para su correcto funcionamiento, luego se parametrizan las entradas de la función principal (main) del algoritmo en la *Figura 25* se observa el código del algoritmo aun sin parametrizar y en la *Figura 26* ya el código parametrizado.

Figura 25. Parte de la función main sin parametrizar

```
26  public class Simulador {
27
28      public static void Simulador(String[] args){
29
30       //Lista de fuentes
31       Senal signal = new Senal(10, 0, 0.16);
32      Coordinate posicion = new Coordinate(1.5, 5.5);
33      Fuente source = new Fuente(posicion, signal);
34      //Creación de la zona de iluminación
35      Coordinate [] puntosZona = new Coordinate[5];
36      puntosZona[0]=new Coordinate(1.5,5.5);
37      puntosZona[1]=new Coordinate(14.5,9.5);
38      puntosZona[2] =new Coordinate(14.5,-4);
39      puntosZona[3]=new Coordinate(1.5,-4);
40      puntosZona[4]=new Coordinate(1.5,5.5);
41
42
43      LinearRing bordeZona=new GeometryFactory().createLinearRing(puntosZona);
44      Polygon poliZona = new GeometryFactory().createPolygon(bordeZona, null);
45      ZonaIluminacion fuenteZI = new ZonaIluminacion(poliZona);
46      source.setZonaDeIluminacion(fuenteZI);//Se le establece la zona de ilumunación a la fuente
47      List<Fuente> transmisores = new ArrayList<Fuente>();
48      transmisores.add(source);
49      //Roi
50      Coordinate[] vertices = new Coordinate[2];
51      vertices[0] = new Coordinate(0, 0);
52      vertices[1] = new Coordinate(12, 14);
53      //Geometría edificio
54      List<Edificio> edificios= new ArrayList<Edificio>();
55      Coordinate [] esqEdif = new Coordinate[5];
56      esqEdif[0]=new Coordinate(5,4.5);
57      esqEdif[1]=new Coordinate(6,6.5);
58      esqEdif[2]=new Coordinate(7.5,6.5);
59      esqEdif[3]=new Coordinate(7,4.5);
60      esqEdif[4]=new Coordinate(5,4.5);
61      LinearRing contorno = new GeometryFactory().createLinearRing(esqEdif);
62      Polygon pooliEdif =  new GeometryFactory().createPolygon(contorno, null);
63      Edificio edificio= new Edificio(pooliEdif);
64      edificios.add(edificio);
65
66      Coordinate [] esqEdif1 = new Coordinate[5];
67      esqEdif1[0]=new Coordinate(6,-1);
68      esqEdif1[1]=new Coordinate(8.5,-1);
69      esqEdif1[2]=new Coordinate(8.5,-3);
70      esqEdif1[3]=new Coordinate(6,-3);
71      esqEdif1[4]=new Coordinate(6,-1);
72      LinearRing contorno1 = new GeometryFactory().createLinearRing(esqEdif1);
73      Polygon pooliEdif1 =  new GeometryFactory().createPolygon(contorno1, null);
74      Edificio edificio1= new Edificio(pooliEdif1);
75      edificios.add(edificio1);
76
```

Figura 26. Función main del algoritmo paramétrizada

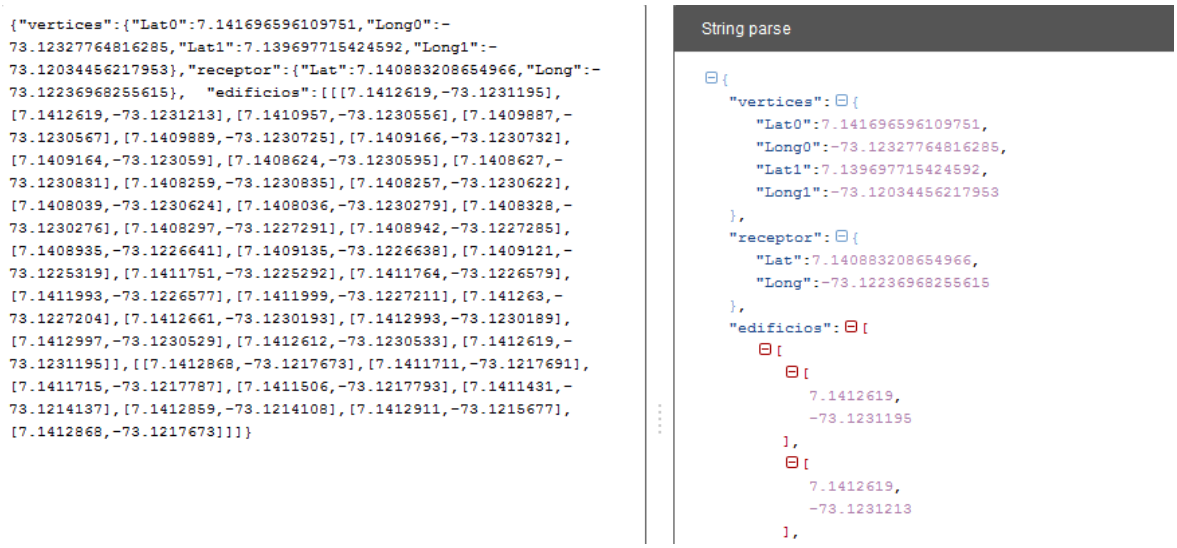
```
30
31  public class Simulacion2 {
32
33          public String simulacion(List<Fuente> fuentes, Coordinate[] vertices,
34                          Receptor receptor, List<Edificio> edificios, List<Arbol> arboles) {
35
36                  Roi region = null;
37                  try {
38                          region = new Roi(vertices, edificios, arboles);
39                  } catch (NoEsPoligonoException ex) {
40                          // Logger.getLogger(Simulador.class.getName()).log(Level.SEVERE,
41                          // null, ex);
42                          return ("No es una region de interes valida");
43                  }
44
45                  ModeloRadioMicroCell prueba = new ModeloRadioMicroCell(fuentes, region,
46                                  -80, 4);
47
48                  // Receptor receptor = new Receptor();
49                  // receptor.setPosicion(new Coordinate(x,y));
50
51                  PixelDeJong pixelInteres = new PixelDeJong(receptor);
52                  double potencia = pixelInteres.contribucionPotencia(
53                                  prueba.getFuentesVirtuales(), region);
54                  return potencia + "";
55                  // System.out.println("potencia de salida= "+potencia);
56
57          }
58  }
```

Esta parametrización se realizó para lograr la correcta comunicación entre los datos enviados por la interfaz gráfica del simulador hacia el web service que se encuentra en la nube realizando los cálculos de la simulación.

Ya lista la parametrización se procede a crear la función principal del web service cuya función es, recibir los datos, enviarlos a las funciones encargadas de procesarlos para que el formato sea compatible con el algoritmo y luego de estos esta misa función envía los datos a la función main del algoritmo quien le regresara un dato de respuesta al web service el cual a su vez envía una respuesta en a la interfaz para que esta sea usada de la forma que se requiera.

Se diseñó la interfaz y el web service para que se comunicaran entre sí en un formato denominado JSON[49] *Figura 27*, el cual funciona en segundo plano y nos permite una ejecución dinámica.

Figura 27. Formato Json y estructura de datos



Descripción: A la izquierda tenemos los datos enviados en formato JSON y a la derecha la estructura de los datos.

Estos datos son recibidos y enviados al esquema principal del web service usando el comando "@Path" para indicarle a la interfaz a donde debe enviar los datos contenidos en el JSON, en la *Figura 28* se observa un Fragmento de la función principal del web service donde se observa el comando PATH.

[49] "Introducing JSON." .

Figura 28. Comando PATH

```
@Path("/postjson")
public class PostJson {

	/**
	 * @param data : entradas necesarias para la simulacion que vienen en un
	 *            string con formato JSON.
	 * @return valor de la potencia en WATTS calculada en el punto en el cual se
	 *         solicita la simulacion.
	 */
	@SuppressWarnings("null")
	@POST
	@Path("/post")
```

Ya que el algoritmo funciona con coordenadas cartesianas se tuvo que realizar una función de conversión de unidades puesto que se planeó que fuera georeferenciado véase la *Figura 29*, teniendo en cuenta que el algoritmo es para áreas propagación de antena no tan amplias o microceldas usamos una conversión de unidades que toma un punto de referencia y toma distancias a partir de ese punto haciendo comparaciones con todos los puntos enviados al algoritmo que hará los cálculos.

Figura 29. Conversión de coordenadas

```
public class Convert {

	/**
	 * @param latitud0 : latitud cero es la latitud tomada del mapa en el punto de la esquina superior izquierda del ROI.
	 * @param longitud0 : longitud cero es la longitud tomada del mapa en el punto de la esquina superior izquierda del roi.
	 * @param latitud1 : latitud uno es la latitud tomada del mapa en el punto de la esquina inferior derecha del roi.
	 * @param longitud1 : longitud uno es la longitud tomada del mapa en el punto de la esquina inferior derecha del ROI.
	 * @param latR : latitud del punto en cual se solicita la simulaci
	 * @param longr : longitud del punto en cual se solicita la simulaci
	 * @return todos los valores antes mencionados en coordenadas X y Y necesarias para la simulaci
	 */
	public static HashMap <String ,  Integer> covertircuadroxcartesiano(double latitud0, double longitud0, double latitud1, double longitud1, double latR, double long

		HashMap <String ,  Integer> respuesta = new HashMap<String, Integer>();

		respuesta.put("x0", 0);
		respuesta.put("y0", 0);



		/*dist2 calcula la distacia a Y; distancia entre (latitud0, longitud0) y (latitud0, longitud1)
		 *dist1 calcula la distacia a X; distancia entre (latitud0, longitud0) y (latitud1, longitud0)
		 */
		int dist1, dist2, distR1, distR2;

		dist2=(int) (Math.abs(longitud1 - longitud0)*100000);
		dist1=(int) (Math.abs(latitud1 - latitud0)*100000);
		distR1=(int) (Math.abs(latR - latitud0)*100000);
		distR2=(int) (Math.abs(longr - longitud0)*100000);

		respuesta.put("x1", dist1);
		respuesta.put("y1", dist2);
		respuesta.put("xr", distR1);
		respuesta.put("yr", distR2);


		return respuesta;


	}
```

Una vez hecha la conversión de unidades el algoritmo de conversión regresa los datos procesados a la función main del webservice y este los envía al algoritmo de simulación para terminar el proceso.

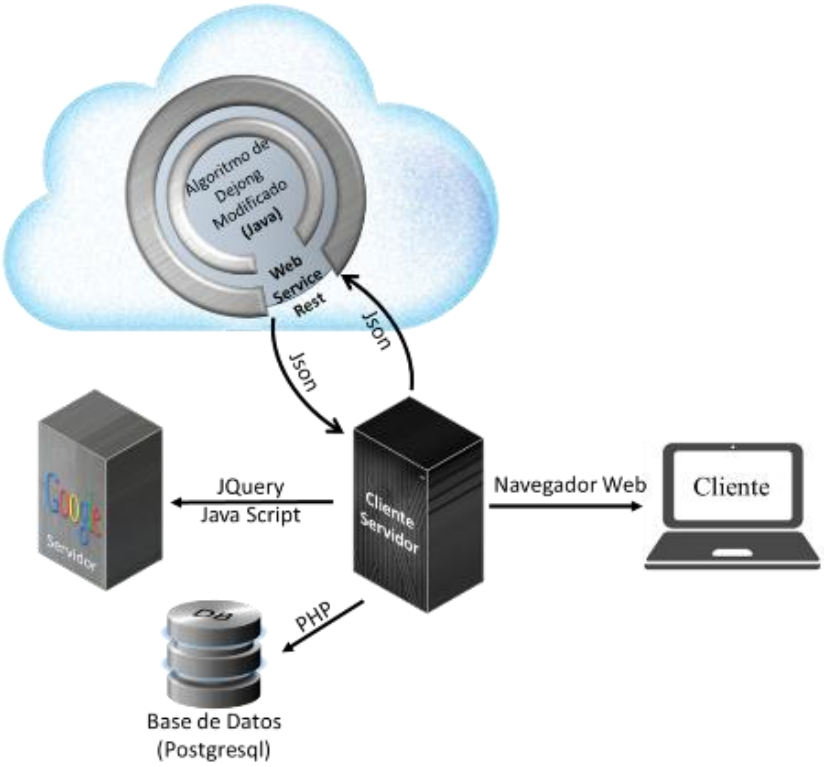
El desarrollo del web service se realizó completamente en JAVA ya que teníamos que tener como base las tecnologías existentes y el algoritmo de simulación fue realizado en este lenguaje de programación.

El paso a seguir fue crear una base de datos que pudiera almacenar toda la información georeferenciada que tenemos, para esto utilizamos el motor de base de datos PostgreSQL y su componente PostGIS el cual es encargado de manejar los datos georeferenciados.

5.1 ARQUITECTURA DE LOS SERVICIOS

Se implementó una arquitectura *cliente servidor* a partir de las tecnologías disponibles, mostrada en la *Figura 30,* esta arquitectura se definió de acuerdo a la necesidad que presentaba el servicio de simulación y con el propósito de adaptar fácilmente los servicios realizados anteriormente por el grupo de investigación RadioGis.

Figura 30. Arquitectura Cliente Servidor



El primer ciclo inicia con el usuario final quien accede a la plataforma por medio de cualquier terminal (computador de escritorio, portátil, móvil) y un navegador web, en este paso el terminal hace las veces de cliente, una vez el cliente realiza una

petición esta acción viaja por la red usando un navegador web hasta llegar al servidor, este servidor se comunica usando unos protocolos y principalmente unos formatos Json que permite la comunicación con el Web Service que mediante entradas paramétrizadas accede al servicio desarrollado en Java “Algoritmo De Jong Modificado”; cuando el algoritmo realiza todo el procesamiento solicitado, envía una respuesta al Web Service, quien se comunica con el servidor y le muestra la respuesta de la petición enviada por el usuario.

En el ciclo dos ocurre que el usuario requiere para la simulación información de un mapa, esta petición viaja por la red utilizando un navegador web hasta llegar al servidor del proyecto, este servidor se comunica usando la API_V03 de Google Maps que permite la comunicación con el servidor de mapas de Google, quien devuelve la respuesta por el mismo medio al usuario final quien está accediendo desde un terminal.

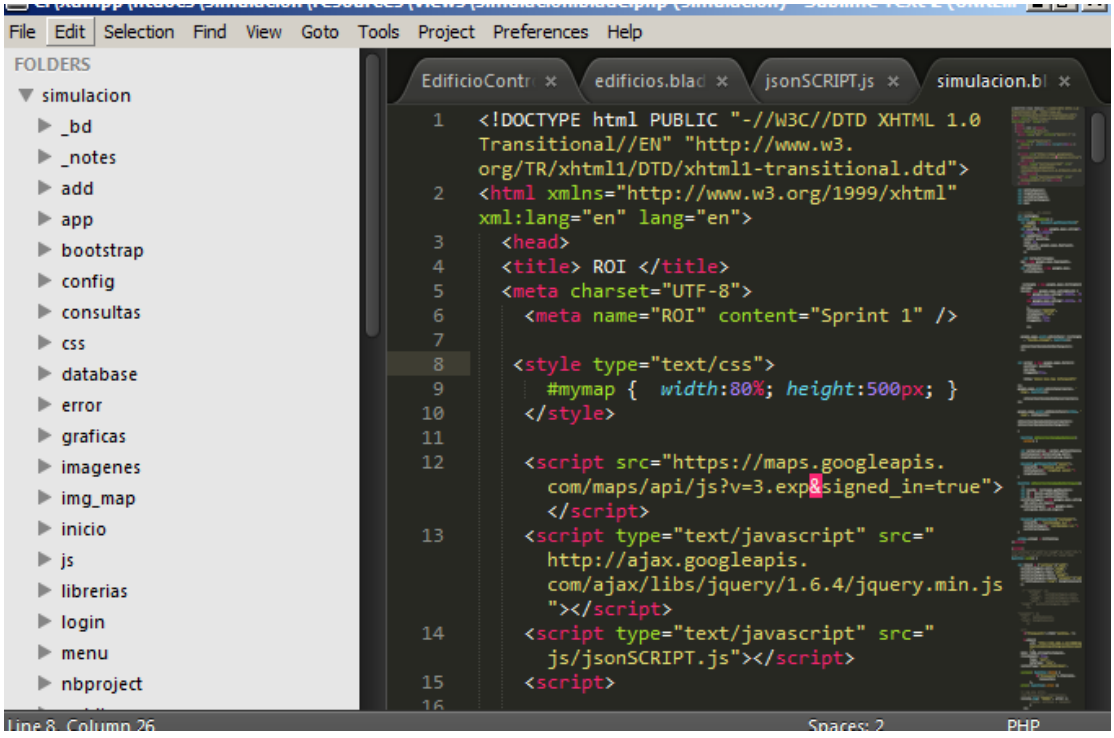
En el ciclo tres el usuario interactúa con la base de datos del proyecto PostgreSQL, ya sea agregando, eliminando o modificando alguno de los componentes del simulador, como lo son los Árboles, Edificios, y Antenas; Esta petición viaja por la red usando un navegador web hasta llegar al servidor del proyecto, quien se comunica con el servidor donde se encuentra alojada la base de datos, este servidor le devuelve la respuesta al usuario quien está haciendo la petición por medio de un terminal que puede ser de bajas prestaciones.

1.1.1 Servicio de mapas: El Api de Google Maps, tiene un papel fundamental para la arquitectura propuesta del servicio de simulación De Jong, ya que este proporciona la vista para el usuario, permitiéndole consultar, agregar, modificar y eliminar los componentes del simulador (Árboles, Edificios, Antenas).

Para el proyecto se utilizó la APIv3 de Google Maps, lo primero que se realizo fue crear en la vista un documento HTML, La API de Google Maps se encuentra alojada en los servidores de Google, para poder cargar la API de Google Maps se debe hacer una referencia desde el archivo HTML hacia el lugar en el que se encuentra ésta. La referencia se ha de incluir en la sección <head> del documento, se carga con el elemento <script>, este elemento tiene dos atributos que se deben utilizar. El primero es el tipo de script que se va a utilizar y el otro es la URL que apunta a la API.

La dirección URL contenida en la etiqueta de script es la ubicación de un archivo JavaScript que carga todos los símbolos y definiciones que se necesita para utilizar la API de Google Maps. Se requiere esta etiqueta de script.

Figura 31. Archivo HTML Conexión APIv3



Se Creó una variable *Figura 32* llamada mapDiv usando el método getElementById() referenciando el nombre creado anteriormente para el mapa “#mymap, también se definieron las coordenadas de la UIS “myLatlng” donde queremos que se ubique el mapa en su inicio.

Figura 32. Propiedades del mapa

```
var latitudsensor;
var longitudsensor;
var punto0rectagulo;
var punto1rectangulo;
var map;


//7.139808, -73.120181
var rectangle;
function initialize() {
  var mapDiv = document.getElementById("
    mymap");
  var myLatlng = new google.maps.LatLng(7.
    139808, -73.120181)
  var mapOptions = {
    center: myLatlng,
    zoom: 17,
    mapTypeId: google.maps.MapTypeId.
      SATELLITE
    };
```

El objeto “mapOptions” mostrado en la *Figura 32* es necesario y define las propiedades de visualización del mapa. Por lo tanto, se crea una variable llamada “options” con las tres propiedades mínimas necesarias para que el mapa funcione:

- **Center**: Define el centro del mapa mediante unas coordenadas.
- **Zoom**: Define el nivel inicial de zoom del mapa. Debe ser un número comprendido entre 1 y 23.
- **MapTypeId**: Define el tipo de mapa que será cargado inicialmente. Los diferentes tipos de mapas se encuentran en el objeto google.maps.MapTypeId

Figura 33. Tipos de mapas en Google Maps

MapTypeId class

`google.maps.MapTypeId` class

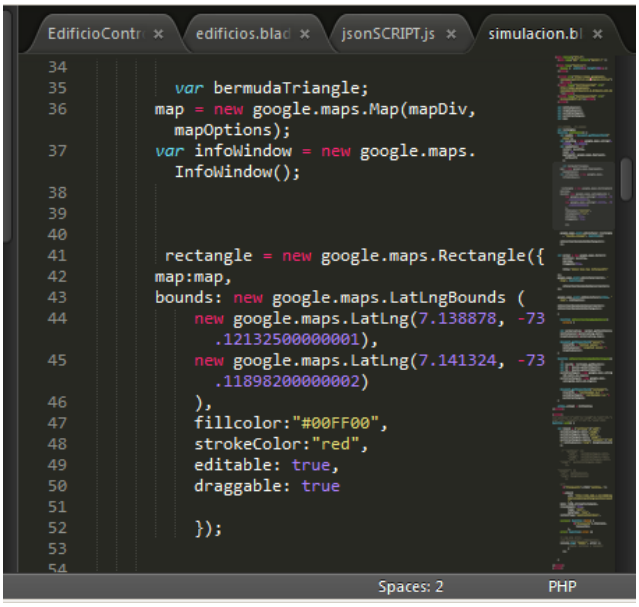
Identifiers for common MapTypes.

Constant	
HYBRID	This map type displays a transparent layer of major streets on satellite images.
ROADMAP	This map type displays a normal street map.
SATELLITE	This map type displays satellite images.
TERRAIN	This map type displays maps with physical features such as terrain and vegetation.

Fuente:https://developers.google.com/maps/documentation/javascript/reference?hl=es#MapOptions

En la *Figura 34* podemos visualizar la creación de la zona de interés (RoI) donde se desea calcular la potencia emitida por una antena, a esta zona de interés se le habilitan propiedades de movimiento por todo el mapa y color.

Figura 34. Zona de interés (RoI)



En la *Figura 35* se evidencia como se creó el marcador para el sensor que recibe la señal proporcionada por la antena dentro del RoI.

Figura 35. Agregación de marcadores

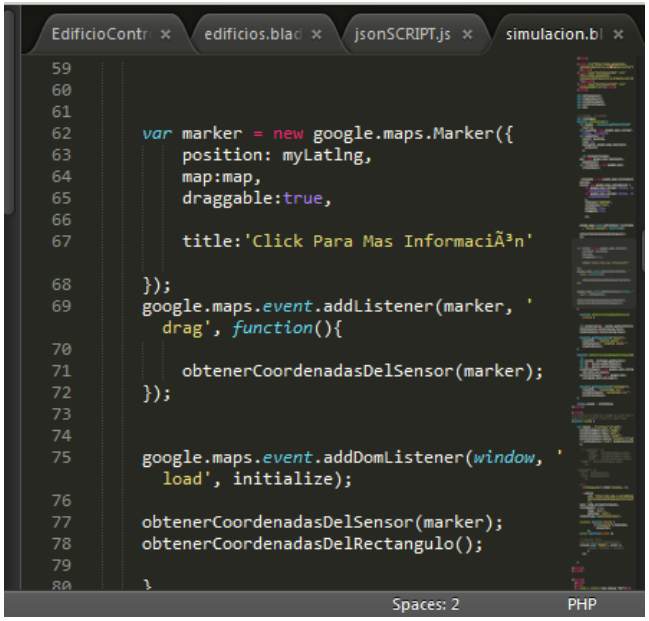


Figura 36. Comunicación entre servidores

```
};*/
  $('#respuesta').html('sending..');

  $.ajax({
        url: 'http://192.168.1.111:8080/WebServiceDeJong/DJong/postjson/post/',
data: JSON.stringify(toSend),
crossDomain: true,
        type: 'post',
        dataType: 'json',
contentType:'application/json',

success: function (data) {
          $('#respuesta').html(data.respuesta);
        },
error: function( error ){

// Log any error.
//$('#respuesta').html(error);
console.log( "ERROR:", error );
        //data: vertices + receptor
        }
  });
```

En la Figura 36 se observa la comunicación por medio de Json entre los servidores web de Google y el del proyecto.

5.1.2 Servicio en la Nube: La implementación del sensor virtual De Jong se realizó bajo el esquema de servicio Sas ya que es el tipo de familia de la computación en la nube más apropiado para la solución que se propuso en el proyecto, este servicio consiste en la entrega de aplicaciones como servicio bajo demanda. Los proveedores de los servicios SaaS pueden tener instalada la aplicación en sus propios servidores web como en este caso en los servidores del grupo de investigación CONUSS permitiendo a los clientes acceder mediante un navegador web.

Contar con los servicios instalados en diferentes maquinas resulta útil, por ejemplo a la hora de realizar actividades como mantenimiento (actualizaciones y backups) de manera eficiente, para implementar estrategias de seguridad y como una medida eficiente para la alta disponibilidad del servicio. Así por ejemplo, la capa de datos se alojó en una maquina con Windows Server, la capa de negocio y de

servicio en una máquina virtual con sistema operativo Linux y sobre el sistema OpenStack que permite darle las capacidades del cómputo en la nube; Las consultas se manejan entre ellas a través de peticiones seguras de consultas a la base de datos. Este modelo tiene varias ventajas como: estabilidad ya que al estar dividido en diferentes máquinas, cada servicio es independiente.

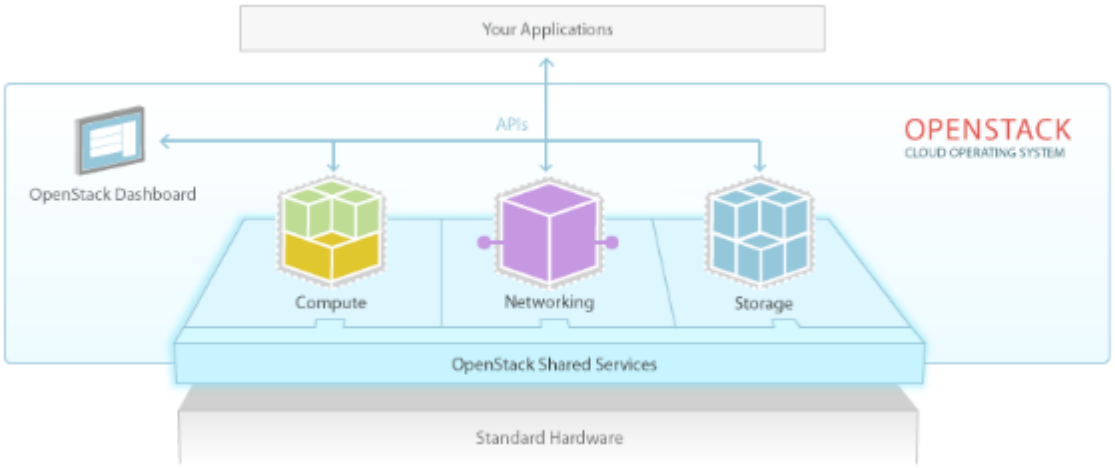
El algoritmo de propagación De Jong requiere de un cómputo avanzado, es por esta razón que la capa de negocio y de servicio fueron implementadas en la nube gracias al sistema OpenStack es un sistema de código abierto para implementaciones en la nube.

El software OpenStack controla grandes pools de cómputo, almacenamiento y recursos de red a través de un centro de datos, gestionado a través de un panel de control o vía OpenStack API. OpenStack trabaja con tecnologías empresarial popular y de código abierto por lo cual es ideal para infraestructura heterogénea. Cientos de las más grandes marcas confían en OpenStack para correr sus negocios cada día, reduciendo costos y ayudando a moverse más rápido. El software es construido por una prospera comunidad de desarrolladores, en colaboración con usuarios.

Este software permite la creación de nubes privadas, hibridas y publicas de cualquier tamaño

Como Funciona:

Figura 37. Arquitectura OpenStack



Fuente: https://www.openstack.org/software/

OpenStack Compute: Provisiona y administra grandes redes de máquinas virtuales. Diseñada con arquitectura flexible para proporcionar libertad al diseñar su nube, sin requisitos de hardware o software y la capacidad de integrar con los sistemas heredados y tecnologías de terceros. Está diseñada para gestionar y automatizar las pools de los recursos de cómputo y puede trabajar con tecnologías de virtualización disponibles ampliamente, así como las configuraciones Bare metal y high-performance computing (HPC).

OpenStack Networking: Permite crear modelos de redes flexibles para adaptarse a las necesidades de las diferentes aplicaciones o grupos de usuarios. Los modelos estándar incluyen redes planas o VLAN para la separación de los servidores y el tráfico además los usuarios pueden crear sus propias redes, control de tráfico y conectar los servidores y los dispositivos a una o más redes. También tiene un marco que permite la extensión de servicios de red adicionales, como los

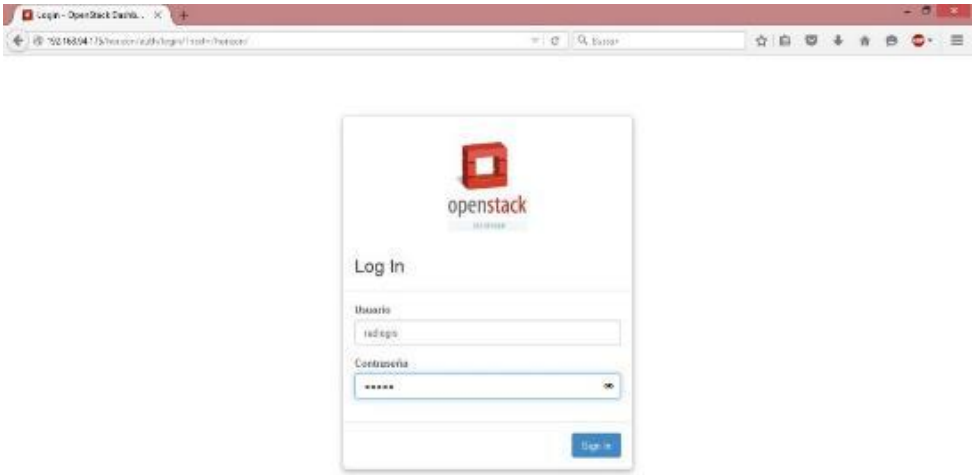
sistemas de detección de intrusos (IDS), balanceo de carga, cortafuegos y redes privadas virtuales (VPN) que se implementan y administran.

OpenStack Storage: Proporciona una totalmente distribuida plataforma de almacenamiento-API accesible que se puede integrar directamente en las aplicaciones o utilizar para copia de seguridad, archivo y conservación de los datos, entre otros usos.

OpenStack Shared Services: Tiene varios servicios compartidos que abarcan los tres pilares de cómputo, almacenamiento y redes, por lo que es más fácil de implementar y operar su nube.

OpenStack Dashboard: El panel de control de OpenStack (dashboard) proporciona al administrador y usuarios una interfaz gráfica para acceder, provisionar y automatizar los recursos basado en la nube. Ademas es fácil incorporar y presentar productos y servicios para terceros, como son la facturación, la monitorización y las herramientas de gestios adicionales.

Figura 38. OpenStack Dashboard



Capacidades de Horizon:

- Permite controlar a los administradores y usuarios su “nivel de computación”, almacenamiento, y recursos de red.
- Como administrador, el panel de control ofrece una vista global del tamaño y estado de la nube. Se pueden crear usuarios y proyectos, asignar usuarios a proyectos y configurar limites en los recursos de esos proyectos.
- El panel de control proporciona a los usuarios un portal “self-service” para provisionar sus propios recursos dentro de los límites establecidos por el administrador.

Crear una maquina en openstack es un proceso eficiente, una vez el administrador de la nube genera un usuario y una contraseña el proceso es el siguiente:

Inicias sesión, en el menú hay que dirigirse al link de instancias, una vez en esa ventana damos click en el botón crear instancia, despliega una ventana emergente en la cual se escoge el sistema operativo y los requerimientos básicos de almacenamiento y memoria RAM.

Una vez echo todo esto podemos ingresar a la maquina a través de una vista de consola de esta que ofrece openstack en la cual a travez de comandos de Linux instalamos lo necesario para la ejecución de nuestro proyecto, al momento de alcanzar el estado óptimo en el cal necesitamos la maquina nos dirgimos nuevamente al menú de openstack y en el menú de instacias vemos que que nuestra maquina en las opciones (un menú desplegable al extremo derecho frente a la descripción de la maquina o “instancia”) encontramos una opción que dice “instantánea de instancia” con esta opción se crea una copia exacta de nuestra

máquina virtual creada anteriormente y podríamos crear más lo cual nos permite una escalabilidad manual del proyecto.

Figura 39. Primera Máquina Virtual De Jong

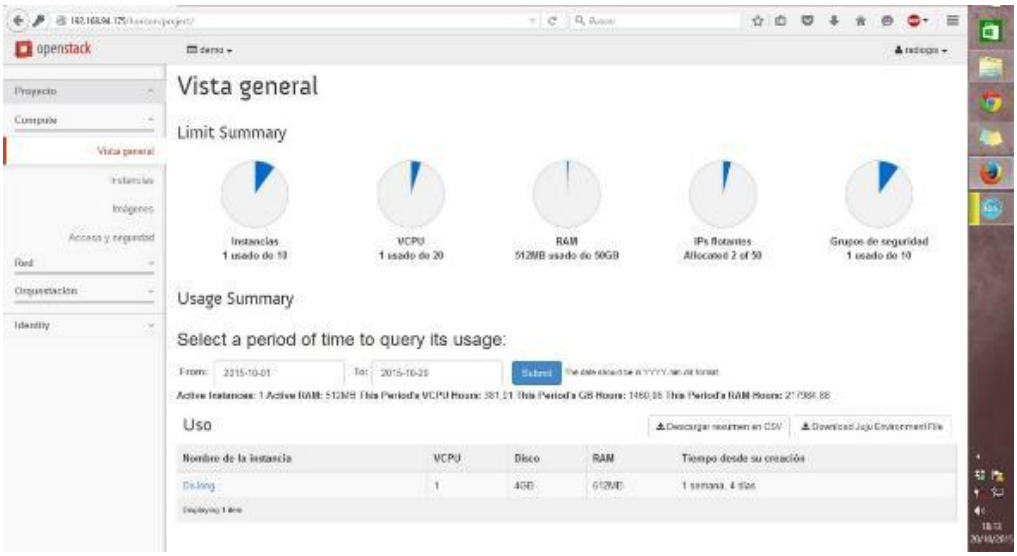


Figura 40. Segunda Máquina Virtual De Jong

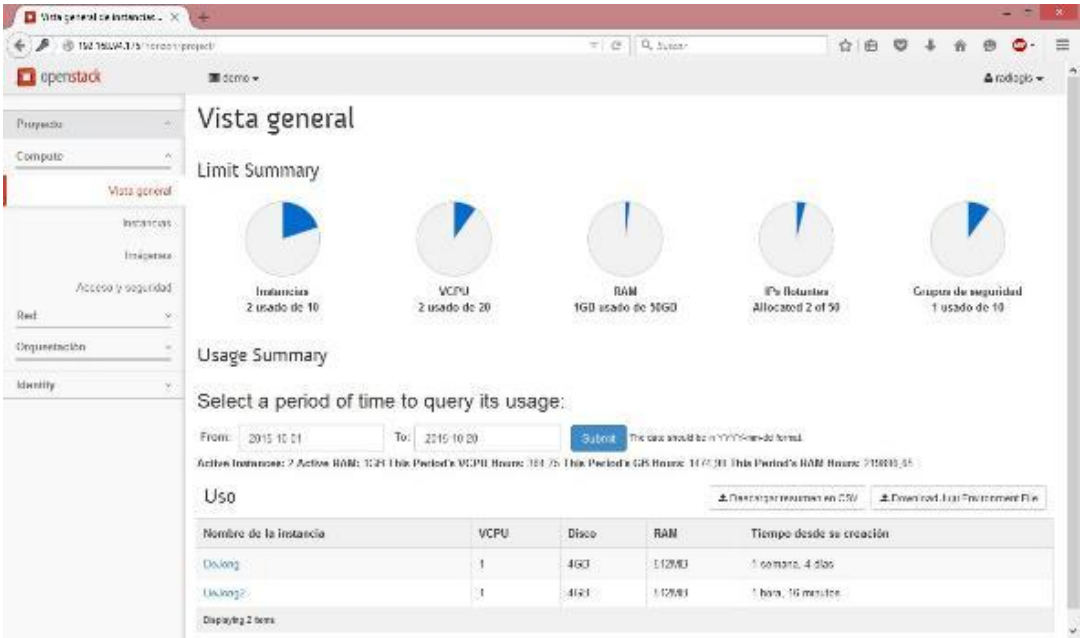
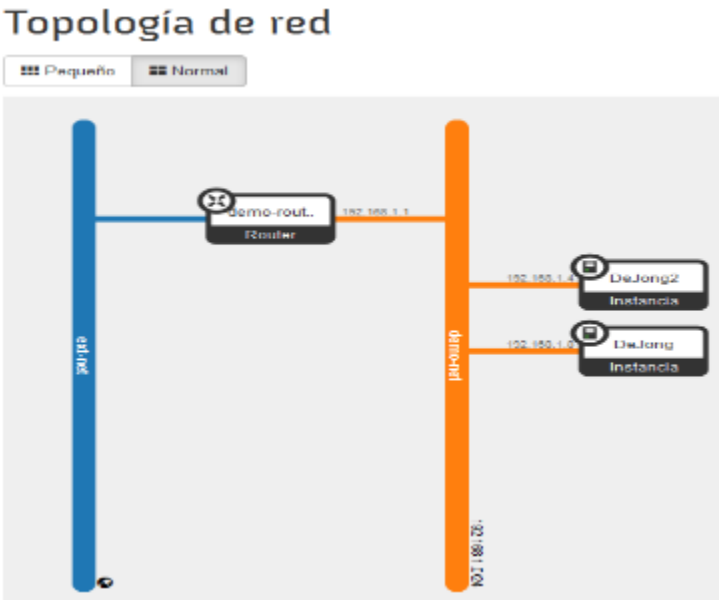


Figura 41. Máquinas Virtuales De Jong



Al tener varias máquinas creadas el openstack administra un balanceo de las peticiones que llegan a las maquinas a travez de una modulo balanceador que tiene una ip externa y realiza el balaceo enviando la petición a la máquina que esté disponible por medio de comunicación interna entre las ips de las máquinas

OpenStack Es considerada una aplicación web extensible que permite a los administradores de la nube y los usuarios controlar sus recursos informáticos, de almacenamiento y de redes, otra característica es que para el administrador, OpenStack Dashboard ofrece una visión global del tamaño y estado de su nube. Puede crear usuarios y proyectos, asignar usuarios a los proyectos y establecer límites en los recursos para esos proyectos, también proporciona a los usuarios un portal de autoservicio para la prestación de sus propios recursos dentro de los límites establecidos por los administradores.

5.2 DESARROLLO DE SENSORES VIRTUALES

Un sensor virtual puede ser visto como un simulador de uno o varios sensores reales, como el sensor que resulta al sumar las capacidades de un sensor real con las de un algoritmo o simplemente como el resultado de unir varios sensores de diferente naturaleza.

Los sistemas de monitoreo del espectro utilizan usualmente equipos de medición, que desde el punto de vista de los servicios equivalen a sensores reales. El algoritmo de propagación De Jong siendo un software también proporciona información similar a la de un equipo de medición (hardware), ya que se utiliza como un servicio de simulaciones virtuales, que permite acceder al algoritmo y realizar simulaciones de mediciones, lo cual podría desempeñar un papel clave en el control y planificación de las redes de UHF.

El uso de los sensores virtuales permite la simulación de mediciones de radiación no ionizantes (NIR), con información detallada de las antenas y geografía en una región de interés (ROI), disminuyendo el costo de los servicios de medición al no depender de equipos físicos. En otros casos los sensores virtuales pueden ser un complemento necesario para cualquier tipo de problema, como en el caso de un médico que considera insuficiente la información que es proporcionada por un electrocardiógrafo y requiere de más información y herramientas para su procesamiento.

La implementación realizada en el presente proyecto para el sensor virtual correspondiente al algoritmo De Jong, presenta una alternativa interesante para el grupo de expertos en sensores. Lo que se hizo fue crear una máquina virtual en la nube dedicada solamente al algoritmo De Jong. A su vez el algoritmo De Jong fue

complementado parametrizando sus entradas para que pudiera tener comunicación con el Web Service tipo REST construido. Como resultado de esta arquitectura, el grupo de expertos en sensores puede seguir participando en la mejora del algoritmo, trabajando sobre la máquina virtual de la misma manera como lo harían en cualquier computador. O bien, podían trabajar en su computador personal para finalmente copiar a la máquina virtual el software de simulación desarrollado. Mientras tanto, el grupo experto en servicios podía trabajar en el desarrollo de un servicio que consume el web service correspondiente al algoritmo De Jong y un servicio web que se utiliza para almacenar el censado de cualquier sensor ya sea virtual o real.

6. RESULTADOS

Los resultados de este proyecto de investigación consisten en el desarrollo de una arquitectura software que permite el cómputo en la nube de algoritmos complejos de simulación, inicialmente se implementó el algoritmo de Dejong, es decir que se convirtió esta aplicación de escritorio en un servicio al cual se puede acceder desde un terminal de pocas prestaciones con acceso a internet, convirtiéndose así en un estrategia de apoyo a la gestión del espectro.

Adicionalmente, por la metodología implementada se mostró que a partir de un algoritmo basado en la lógica de un sensor real, se puede construir un sensor virtual, por lo cual se puede decir que los sensores virtuales y reales tienen el mismo comportamiento en el modelo propuesto para el usuario, facilitando el diseño e implementación de servicios complejos.

Bajo esta idea de estrategia de implementación de los algoritmos de radiopropagación como sensores virtuales se realizó una ponencia en uno de los congresos sobre comunicación y computación más importantes del país, el 8° IEEE COLCOM, realizado del 13 al 15 de mayo de 2015 en Popayán (Cauca). La ponencia fue premiada con la publicación del Artículo Anexo 2 "*Implementation of DeJong radio propagation algorithm as a virtual sensor for cloud services to enhance radio spectrum management"* [30], en la IEEEXplorer, una de las bibliotecas virtuales más grandes del mundo.

6.1 HISTORIAS DE USUARIO

La metodología Scrum descrita anteriormente nos propone crear una lista de requisitos llamados también “historias de usuarios”, en estas historias de usuario se describe lo que los usuarios finales deberían hacer al finalizar las etapas de desarrollo en los “Sprint”. A Continuación se describen las acciones que se pueden realizar en la plataforma por parte de los dueños del servicio.

6.1.1 Sprint 1 El usuario básico puede

1. Entrar a la plataforma UISpectrum (http://radiogis.uis.edu.co/uispectrum/login/), mediante cualquier terminal.
2. Ingresar al módulo de Simulación De Jong, y allí encontrar un mapa con unos objetos de interés fijos: árboles, edificios.
3. mover el sensor
4. puede variar la ROI
5. oprime consultar potencia y obtiene en forma de texto la potencia medida en el punto donde está el sensor

6.1.2 Sprint 2 El usuario administrador puede,

Servicio de edición de mapas

1. Entrar a la plataforma UISespectro, mediante una página web
2. ingresa al módulo de Simulación De Jong. Allí encuentra un mapa que puede editar para agregar o quitar objetos de interés como: árboles, edificios y antenas
3. Puede guardar los objetos de interés introducidos por el usuario
4. Puede invocar objetos de interés previamente guardados.
5. calcular la potencia en un punto con todos los objetos de interés.

6.1.3 Sprint 3 El usuario puede,

Estando en el módulo de simulación De Jong, guardar muchos puntos (el sensor ubicado en diferentes posiciones) en una base de datos.

6.1.4 Sprint 4 El usuario puede: hacer lo mismo que el sprint 3, pero con la información aportada por otro tipo de sensores.

6.1.5 Sprint 5, Siguientes mejoras Para la base de datos y el webservice, que contemplen lo siguiente:

- las mediciones pueden ser multidimensionales, es decir, que una medición puede tener varios valores separados por símbolos (prevemos que sean espacios entre los valores, pero punto y coma entre las filas). La idea es que si hay varios sub-sensores, como los de la nariz electrónica, en realidad se trata de un solo sensor, que es la nariz, solo que la medición está compuesta de varios valores no solo porque hay varios sub-sensores, sino porque el resultado de la medición.
- Se incluye la posibilidad de crear campañas. La campaña lleva información sobre varias cosas, entre otras el tiempo de muestreo (el tiempo entre muestra y muestra).
- Se incluye la casilla de etiqueta opcional (para etiquetas separadas por comas). Para no modificar el webservice, se puede usar la casilla “banda” y simplemente cambiarle de nombre para que se llame etiqueta.
- Se conserva todo lo demás
- Probar que todo funciona

6.2 ESQUEMA PRUEBAS FUNCIONALES

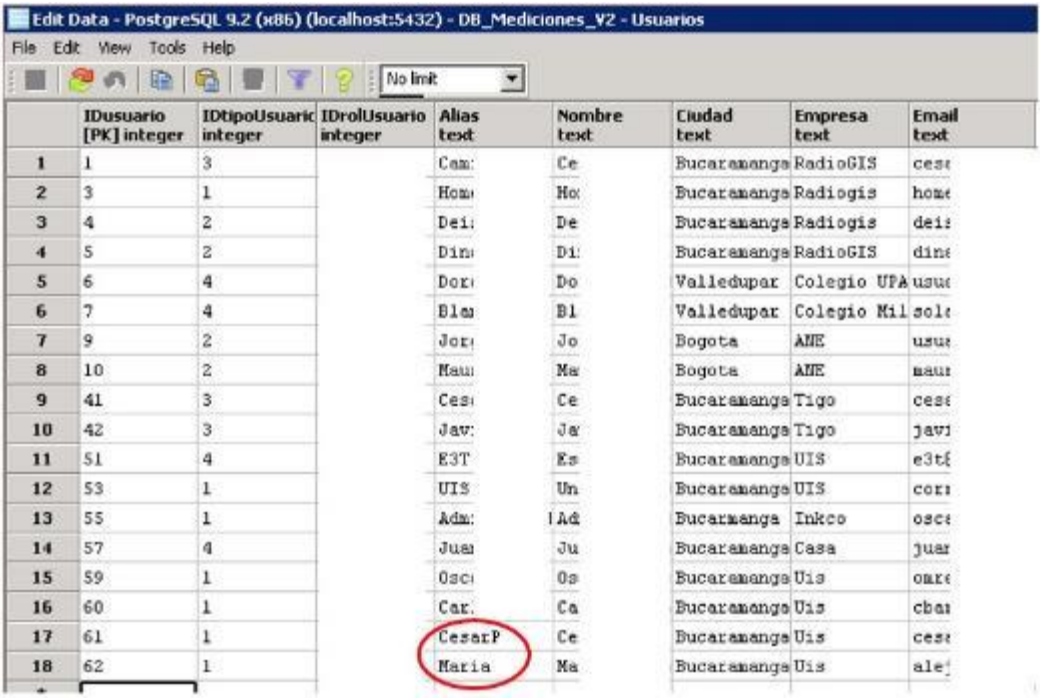
6.2.1 Módulo de usuarios

Tabla 5. Prueba funcional módulo Usuario

Número	Acción	Resultado	Aprobado
1	Registrar un usuario administrador	Se pudo registrar el usuario.	Si
2	Iniciar sesión con el usuario	Se ingresó a la plataforma con el usuario creado	Si
3	Eliminar usuario previamente creado	Se pudo eliminar el usuario	Si

Para registrar un usuario de cualquier tipo lo debe hacer el administrador de la plataforma, este debe dirigirse a la base de datos Postgresql y asignarle el tipo de valor según sus necesidades; Para que sea un usuario administrador debe tener el tipo de valor 1, en la *Figura 42* se evidencia la creación de dos usuarios administradores.

Figura 42. Creación de usuarios desde la base de datos



	IDusuario [PK] integer	IDtipoUsuario integer	IDrolUsuario integer	Alias text	Nombre text	Ciudad text	Empresa text	Email text
1	1	3		Cam:	Ce	Bucaramanga	RadioGIS	ces
2	3	1		Hom:	Ho	Bucaramanga	Radiogis	hom
3	4	2		Dei:	De	Bucaramanga	Radiogis	dei
4	5	2		Din:	Di	Bucaramanga	RadioGIS	din
5	6	4		Dor:	Do	Valledupar	Colegio UPA	usu
6	7	4		Ble:	Bl	Valledupar	Colegio Mil	sol
7	9	2		Jor:	Jo	Bogota	ANE	usu
8	10	2		Mau:	Ma	Bogota	ANE	mau
9	41	3		Ces:	Ce	Bucaramanga	Tigo	ces
10	42	3		Jav:	Ja	Bucaramanga	Tigo	jav
11	51	4		E3T	Es	Bucaramanga	UIS	e3t
12	53	1		UIS	Un	Bucaramanga	UIS	cor
13	55	1		Adm:	Ad	Bucarmanga	Inkco	osc
14	57	4		Jua:	Ju	Bucaramanga	Casa	jua
15	59	1		Osc:	Os	Bucaramanga	Uis	omr
16	60	1		Car:	Ca	Bucaramanga	Uis	cba
17	61	1		CesarP	Ce	Bucaramanga	Uis	ces
18	62	1		Maria	Ma	Bucaramanga	Uis	ale

En la *Figura 43* se evidencia el usuario administrador Cesar P [Administrador] ingresando a la plataforma.

Figura 43. Ingreso a la plataforma con el usuario creado

También existe un formulario para usuarios particulares que estén interesados en formar parte de la plataforma, se puede hacer el registro en la opción solicitar

nuevo usuario mostrada en la *Figura 44*, esto lo remitirá a un formulario de Google y posteriormente se verificara su acceso.

Figura 44. Solicitud de usuario particular

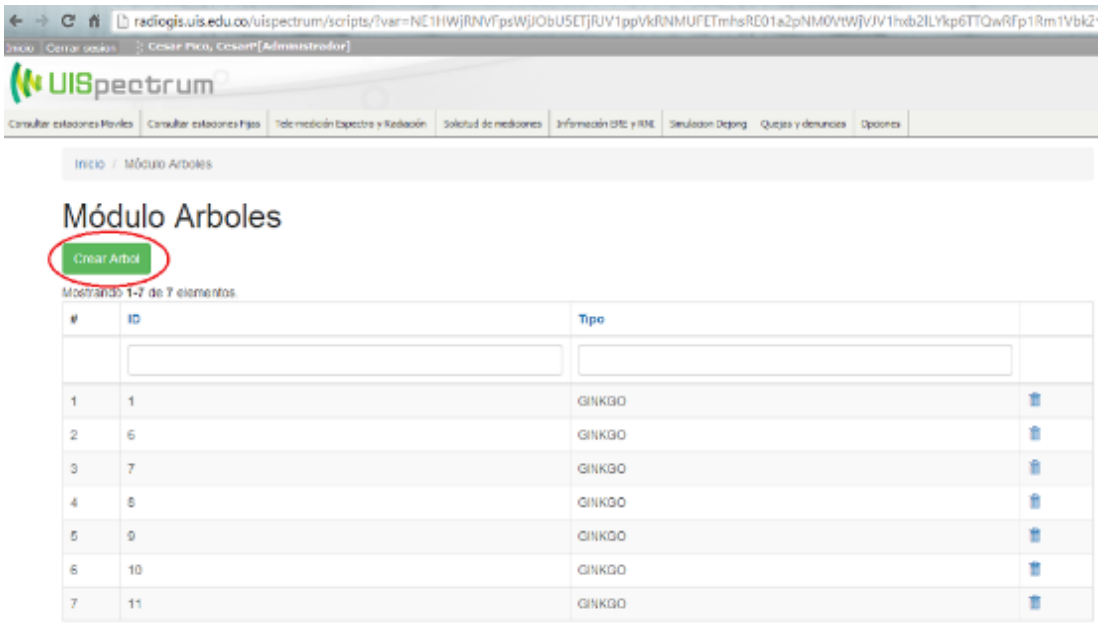
6.2.2 Módulo de Árboles: Este Modulo es importante para el cálculo de la potencia emitida por una antena, ya que la onda sufre un efecto *scattering* cuando choca con las ramas de los árboles que se encuentran en su camino.

Tabla 6. Prueba funcional módulo arboles

Número	Acción	Resultado	Aprobado
1	Crear un árbol	Se crean árboles satisfactoriamente	Si
2	Modificar posición del árbol	Se cambió la posición de los arboles	Si
3	Eliminar árbol de la base de datos.	Se eliminaron los árboles satisfactoriamente	Si

Accedemos al Módulo Árboles véase la *Figura 45*, en la cual podemos observar los árboles almacenados en la base de datos, seleccionamos la opción de crear árbol; en esta misma lista podemos eliminar uno o varios árboles.

Figura 45. Módulo arboles

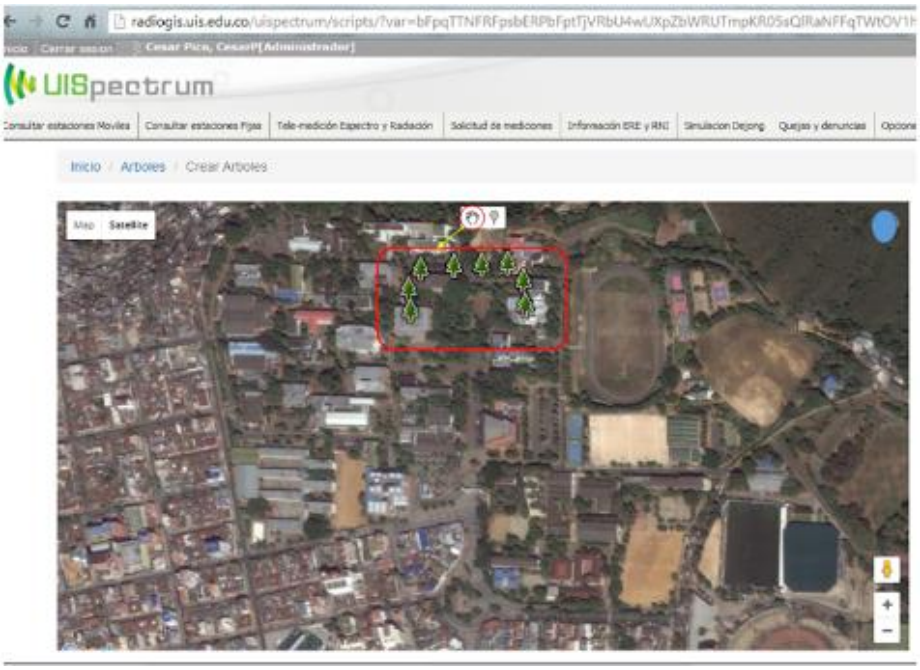


Procedemos agregar un marcador con la ubicación deseada para el árbol como se muestra en la *Figura 46*.

Figura 46. Creación de árboles en la plataforma

La ubicación de los árboles se puede modificar, con la opción de arrastre mostrada en la *Figura 47*, dentro del módulo árboles.

Figura 47. Modificación a la posición de los arboles



6.2.3 Módulo de Antenas Las antenas son las encargadas de emitir la radiación hacia el sensor, estas antenas tiene varios parámetros como su amplitud, fase y longitud de inda que varían de acuerdo al tipo de antena.

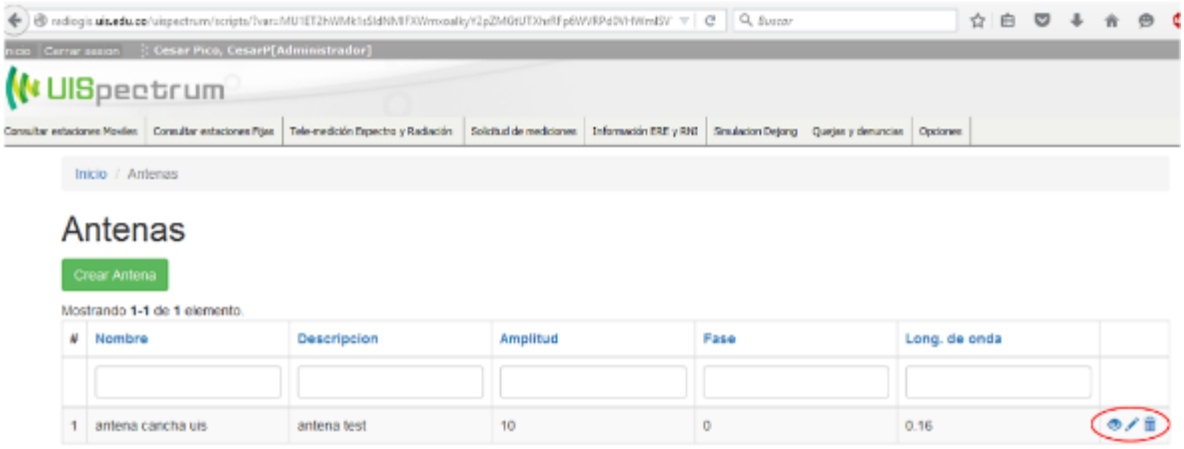
Tabla 7. Prueba funcional módulo antenas

Número	Acción	Resultado	Aprobado
1	Crear una antena	Se crean antenas satisfactoriamente	Si
2	Modificar antena	Se modifican antenas satisfactoriamente	Si
3	Eliminar Antena	Se eliminan antenas satisfactoriamente	Si

En el modulo de antenas se tiene la opcion de buscar especificamente una antena creada anteriormente, en la *Figura 48* se evidencia solo una antena almacenada

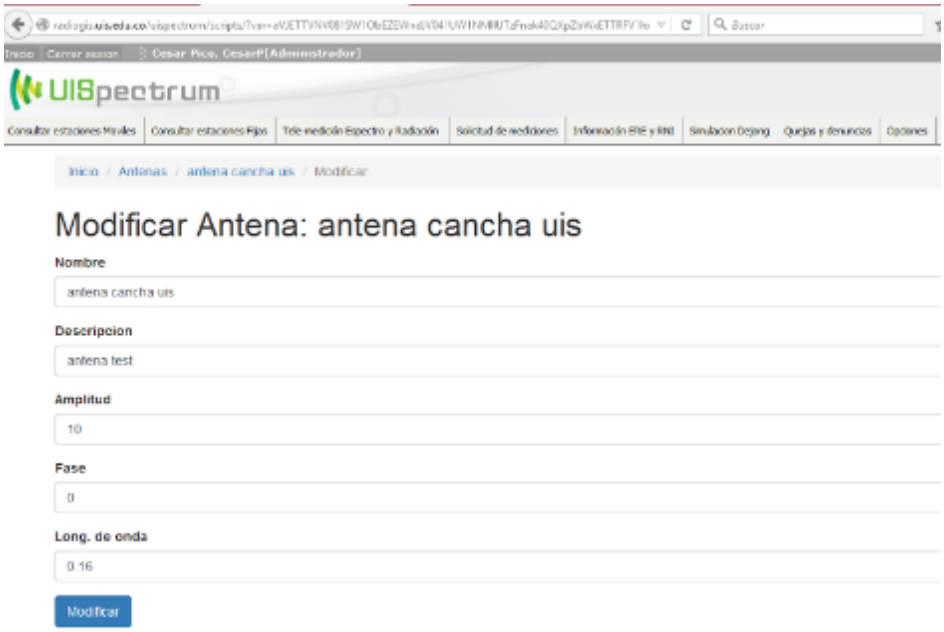
en la base de datos y en el circulo rojo se muestran las opciones habilitadas para la antena como visualizar la antena, modificarla y eliminarla.

Figura 48. Módulo antenas



Al momento de modificar una antena se deben introducir parametros especificos, que deben ser suministrados por expertos especificamente en el campo del espectro y la radiación.

Figura 49. Campos a modificar en una antena



6.2.4 Modulo edificios Los edificios son un factor clave en la medición de la radiopropagación, ya que estos sufren varios efectos sobre la onda de radiación emitida por una antena.

Tabla 8. Prueba funcional módulo edificio

Número	Acción	Resultado	Aprobado
1	Crear un edificio	Se crean los edificios satisfactoriamente	Si
2	Modificar edificios	Se modifican los edificios satisfactoriamente	Si
3	Eliminar edificios	Se eliminan los edificios satisfactoriamente	Si

Para crear un edificio se debe trazar el contorno del edificio a agregar, sobre el mapa. Ver *Figura 50,* estas coordenadas son guardados automáticamente en la base de datos.

Figura 50. Creación de edificios

Para eliminar un edificio se cuenta con dos maneras para hacerlo:

- La primera forma es desde la vista que provee el módulo de Edificios *Figura 51,* donde se puede observar, modificar y eliminar un edificio almacenado.

Figura 51. Eliminación de un edificio desde el módulo arboles

- La segunda forma de eliminar un edificio, *Figura 52* es mediante el módulo de crear edificios, con click derecho sobre lo que se desea eliminar.

Figura 52. Eliminación de un edificio desde el mapa

6.2.5 Módulo de simulación Se realizaron cambios para que el algoritmo procesara elementos pertenecientes a la geografía colombiana, específicamente en la Universidad Industrial de Santander. En la *Figura 53,* podemos observar el auditorio Luis A Calvo, el edificio Capruis, el edificio de la escuela Ingenieria mecanica, y algunos otros que se encuentran dentro del campus Universitario. A partir de los árboles, edificios y antenas existentes en la zona de interés, el algoritmo De Jong Calcula la potencia recibida por un sensor.

Figura 53. Calculo de la potencia

6.2.6 Módulo de sensores - De Jong, Modelo a seguir.

Tabla 9. Módulo Sensores

Número	Acción	Resultado	Aprobado
1	Crear un sensor	Se crean los sensores satisfactoriamente	Si
2	Modificar un sensor	Se modifican sensores satisfactoriamente	Si
3	Eliminar un sensor	Se eliminan sensores satisfactoriamente	Si

Se puede observar que la creación de sensores virtuales y reales tienen el mismo procedimiento, Figura 54.

Figura 54. Creación de sensores

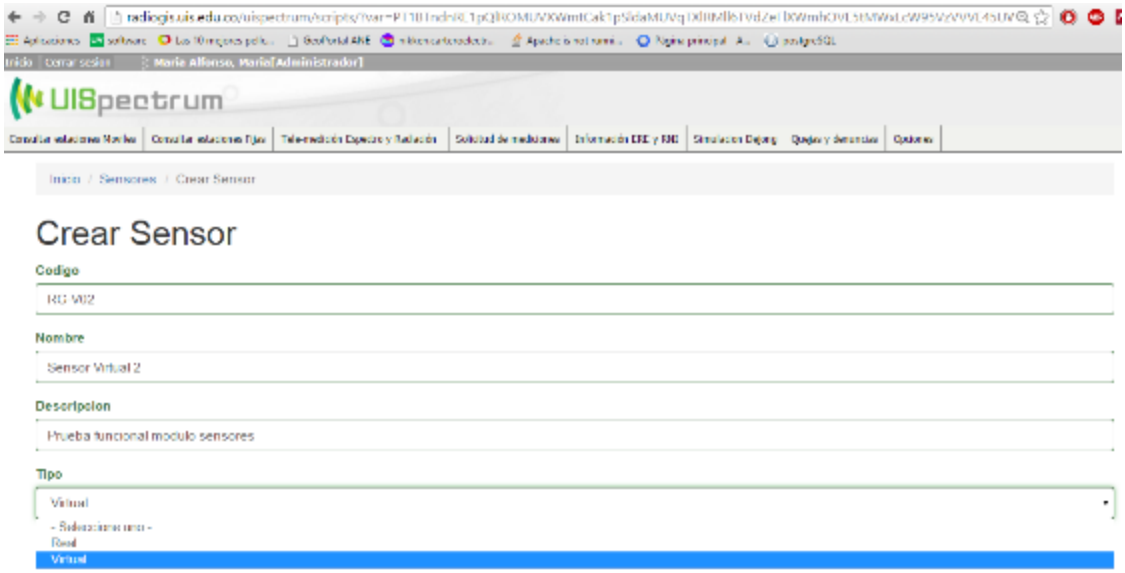


Figura 55. Búsqueda de sensores almacenados

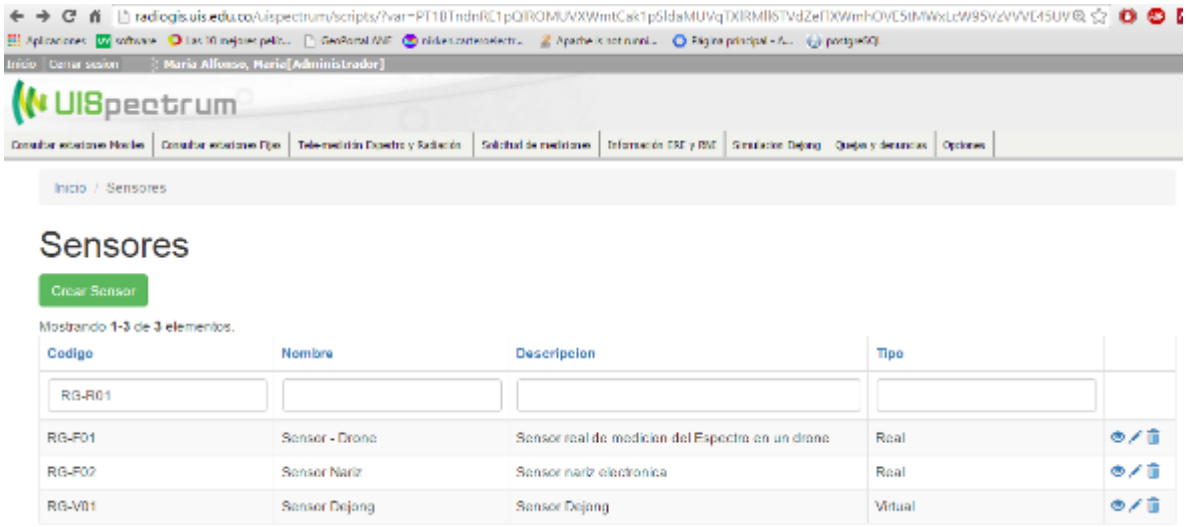


Figura 56. Actualizar o eliminar un sensor

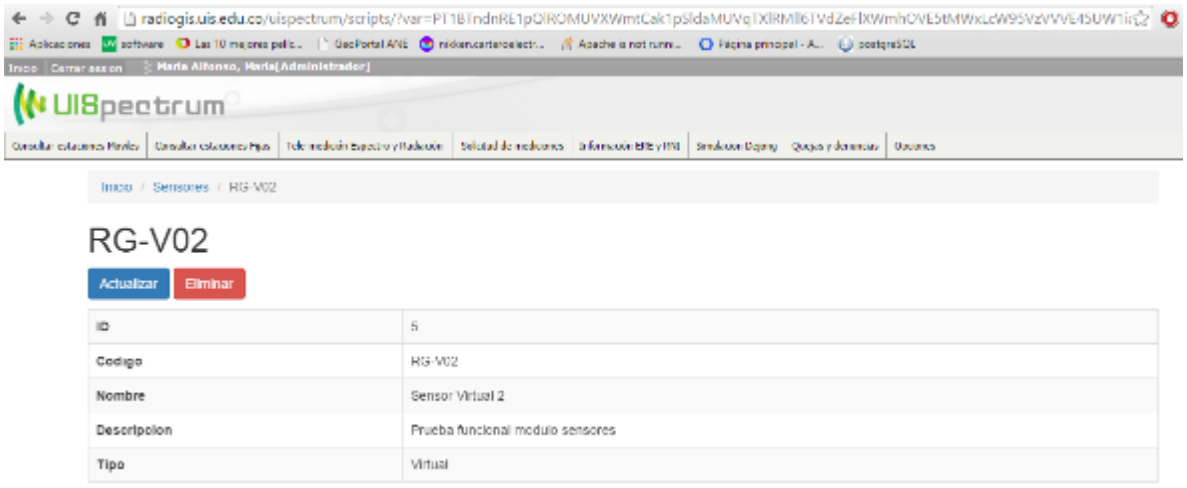
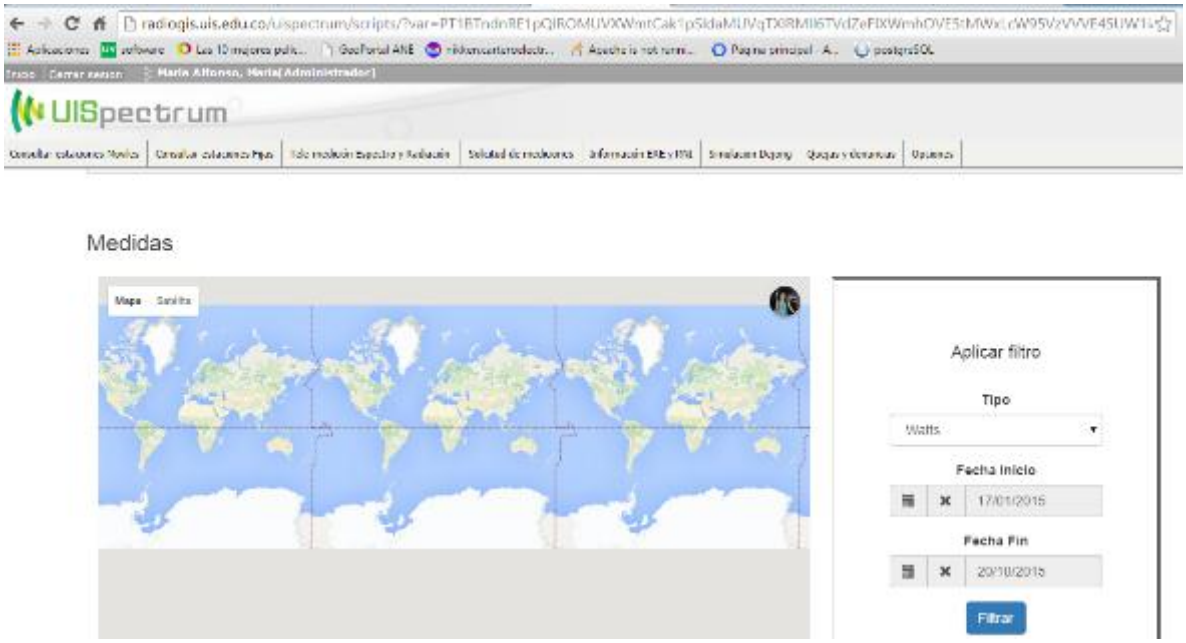


Figura 57. Consulta de medidas por tipo de sensor



Con este módulo de sensores se garantiza que cualquier sensor sea Real o Virtual pueda implementarse en esta plataforma, desde y cuando tenga el mismo comportamiento de los sensores ya agregados.

En el momento se tiene funcionando el algoritmo de propagación Dejong con el comportamiento de un sensor virtual que apoya ciertas tareas de gestión del ERE como: realizar consideraciones sobre lo que puede ocurrir si se agrega o si se retira una o varias estaciones bases de ciertos sitios; y comparar mediciones con simulaciones para identificar emisiones clandestinas. Además que los sensores virtuales de medición del ERE son claves para predecir medidas en condiciones variantes, como el terreno, la entrada de nuevas tecnologías, la configuración de antenas, etc.

Esto es una acción muy importante para futuros servicios que se deseen beneficiar de este modelo propuesto que ofrece muchas ventajas como bajos costos de implementación, comodidad para los usuarios, alto desempeño.

pero no solo se piensa abarcar esta área ya que por la metodología implementada se mostró que a partir de un algoritmo basado en la lógica de un sensor real, se puede construir un sensor virtual, por lo cual se puede decir que los sensores virtuales y reales tienen el mismo comportamiento en el modelo propuesto para el usuario, facilitando el diseño e implementación de servicios complejos, no solo para el area de las telecomunicaciones, sino para otras areas importantes de la industria como por ejemplo la agricultura, la energía, ecología, planeación vial (sensores de carbono en la zona), medicina, etc.

Aproximadamente hace 15 años se imaginaba un mundo donde todo estuviera conectado, hoy es posible, prácticamente el 100% de los objetos están conectados gracias al internet de las cosas.

Se ha llegado al momento en que el mundo digital tiene una influencia en el mundo real y por ejemplo en este caso se puede observar con los sensores virtuales que se construyen basados en la lógica de los sensores reales, es por esta razón que pueden ser tratados de igual a igual y este proyecto es una clara idea de esto, teniendo en cuenta que los sensores virtuales y los sensores reales manejan con gran similitud los datos, se puede seguir generando sensores virtuales que complementen los datos de los sensores reales o que además puedan facilitar la toma de datos en algunos momentos donde los sensores reales no puedan llegar como por ejemplo al "futuro" (proyecciones).

En la universidad industrial de Santander se ha llegado este momento, ya que se está manejando los datos de sensores virtuales como el simulador DeJong y sensores reales como las narices electrónicas en la misma base de datos, sensores reales y virtuales se encuentran al mismo nivel en datos.

7. CONCLUSIONES

Se comprobó que es posible tomar un algoritmo o simulador y convertirlo en un sensor virtual al llevarlo a un web service en la nube que permite la comunicación entre sus diferentes componentes, esto permite hacer accesible el uso de cualquier sensor real para cualquier usuario sin la necesidad de equipos de cómputo costoso; se seguirán agregando sensores a la base de datos con la meta de que en un futuro cercano se tenga acceso a la información de un regimiento de sensores que pueda ayudar a llevar una vida mejor y tranquila.

Se demostró la viabilidad y eficiencia de una arquitectura cliente servidor, con procesamiento en la nube para un algoritmo suficientemente complejo como es el algoritmo de propagación De Jong, convirtiendo una aplicación de escritorio en un servicio que se obtiene desde un terminal de pocas prestaciones con acceso a internet.

Los sensores virtuales y reales tienen el mismo comportamiento en el modelo propuesto para el usuario, facilitando el diseño e implementación de servicios complejos.

Los sensores virtuales de medición del ERE pueden convertir una terminal de bajas prestaciones como un celular o una Tablet en medidores de niveles de potencia o incluso de RNI claves para predecir medidas en condiciones variantes, como el terreno, la entrada de nuevas tecnologías, la configuración de antenas; todo esto mediante una terminal de

Para la implementación en la nube del sensor virtual Dejong se utilizó un sistema creado para manejar diferentes instancias en la nube con fines específicos, demostrando una ventaja de administración cuando se presentan posibles fallas, permitiendo rastrearlas de una manera más eficiente.

Trabajar con la metodología ágil Scrum fue un acierto porque permitió tener una comunicación más fluida con los miembros del equipo donde se priorizaban tareas cada semana y al tener que mostrar resultados tangibles en cada iteración, el cliente en este caso los profesores se sentían más involucrados e identificados con el producto.

Cuando se unen expertos en diferentes áreas, cada uno de ellos aprende más y se pueden construir soluciones más robustas con las cuales se beneficiaran a ambas partes.

BIBLIOGRAFÍA

ALAN KAY y la metáfora del escritorio. [Online]. Available: http://www.adelal.com/nOproblemO/LVR/Cap6.htm. [Accessed: 12-Sep-2015].

ALBERTO URUEÑA E. V., FERRARI Annie, BLANCO David, Cloud Computing. Retos y Oportunidades. España, 2012.

ATHANASIADOU G. E., NIX A. R., and MCGEEHAN J. P., “A microcellular ray-tracing propagation model and evaluation of its narrow-band and wide-band predictions,”

AYMERICH F. M., FENU G., and SURCIS S., “An approach to a cloud computing network,” 1st Int. Conf. Appl. Digit. Inf. Web Technol. ICADIWT 2008, pp. 113–118, 2008.

BOADA H. O., FLOREZ M. G., ARIZA J. D. R., MUNOZ M. A. A., MARTINEZ C. A. P., and FORERO C. A., “Implementation of DeJong radio propagation algorithm as a virtual sensor for cloud services to enhance radio spectrum management,” in IEEE Colombian Conference on Communication and Computing (IEEE COLCOM 2015), 2015, pp. 1–7.

BUYYA R., YEO C. S., VENUGOPAL S., BROBERG J., and BRANDIC I., “Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility,” Futur. Gener. Comput. Syst., vol. 25, no. 6, pp. 599–616, Jun. 2009.

CASTEJÓN GARRIDO J. S., “Arquitectura y diseño de sistemas web modernos,” Rev. Ing. Informáctica del CIIRM, pp. 1–6, 2004.

CREATIVAWEB Servicios en la Nube para Pymes y sus Beneficios | Creativa Web®. [Online]. Available: http://www.creativaweb.co/servicios-en-la-nube-para-pymes/. [Accessed: 13-Sep-2015].

DEJONG Y. L. C. and HERBEN M. H. A. J., “A Tree-Scattering Model for Improved Propagation Prediction in Urban Microcells,” IEEE Trans. Veh. Technol., vol. 53, no. 2, pp. 503–513, Mar. 2004.

GLOBAL I. B. M. and SERVICES T., “máquinas virtuales en la nube de IBM,” pp. 1–4.

GONZAGA J. C. B., MELEIRO L. a C., KIANG C., and MACIEL FILHO R., “ANN-based soft-sensor for real-time process monitoring and control of an industrial polymerization process,” Comput. Chem. Eng., vol. 33, pp. 43–49, 2009.

IEEE J. Sel. Areas Commun., vol. 18, no. 3, pp. 322–335, Mar. 2000.

JONG Y. Y., “Measurement and modelling of radiowave propagation in urban microcells,” 2001.

KOULOPOULOS T. M., Cloud Surfing: A New Way to Think About Risk, Innovation, Scale and Success. 2012.

KRASNER G. E. and POPE S. T., “A cookbook for using the model-view controller user interface paradigm in Smalltalk-80,” J. Object-Oriented Program., vol. 1, no. 3, pp. 26–49, Aug. 1988.

LEE B. M. and KANG U. G., “Virtual Sensor for Diabetes Meter in U-Health Service,” vol. 6, no. 6, pp. 87–96, 2014.

MARSET R. N., RESTRESTvsvsWeb ServicesWeb Services. 2007.

MEJÍA Y. PINEDA and M., Implementación en java de un algoritmo de radiopropagación basado en el modelo de Jong y los requisitos de propagación de los sistemas IMT-ADVANCED. 2011.

MELGAREJO Y. H. M., BOADA H. O., and ALHUCEMA M. D. P., “A model for urban microcell radio propagation prediction focused on reliable implementation,” 2012 IEEE Colomb. Commun. Conf. COLCOM 2012 - Conf. Proc., no. 1, 2012.

MELL P. and GRANCE T., “The NIST Definition of Cloud Computing Recommendations of the National Institute of Standards and Technology,” Nist Spec. Publ., vol. 145, p. 7, 2011.

MILLER M., Cloud Computing: Web-Based Applications That Change the Way You Work and Collaborate Online. 2008.

P. Modelo-vista-controlador, “Patrón Modelo-Vista-Controlador.,” vol. 11, no. 1, pp. 47–57, 2012.

PINEDA ALHUCEMA M. D., ORTEGA BOADA H., and NAVARRO CADAVID A., “Modified Version of de Jong Radio Propagation Model,” Int. Informatics Telecommun. Symp. I2TS2010, pp. 141–147, 2010.

POSTGRESQL “Sobre PostgreSQL.” [Online]. Available: http://www.postgresql.org.es/sobre_postgresql. [Accessed: 06-Oct-2015].

POSTGRESQL: Advantages. [Online]. Available: http://www.postgresql.org/about/advantages/. [Accessed: 06-Oct-2015].

RIZK K., WAGEN J.-F., and GARDIOL F., “Two-dimensional ray-tracing modeling for propagation prediction in microcellular environments,” IEEE Trans. Veh. Technol., vol. 46, no. 2, pp. 508–518, May 1997.

SHENG Q. Z., QIAO X., VASILAKOS A. V., SZABO C., BOURNE S., and XU X., “Web services composition: A decade’s overview,” Inf. Sci. (Ny)., vol. 280, pp. 218–238, 2014.

SOFTENG, “Proceso y Roles de Scrum.” [Online]. Available: https://www.softeng.es/es-es/empresa/metodologias-de-trabajo/metodologia-scrum/proceso-roles-de-scrum.html. [Accessed: 13-Oct-2015].

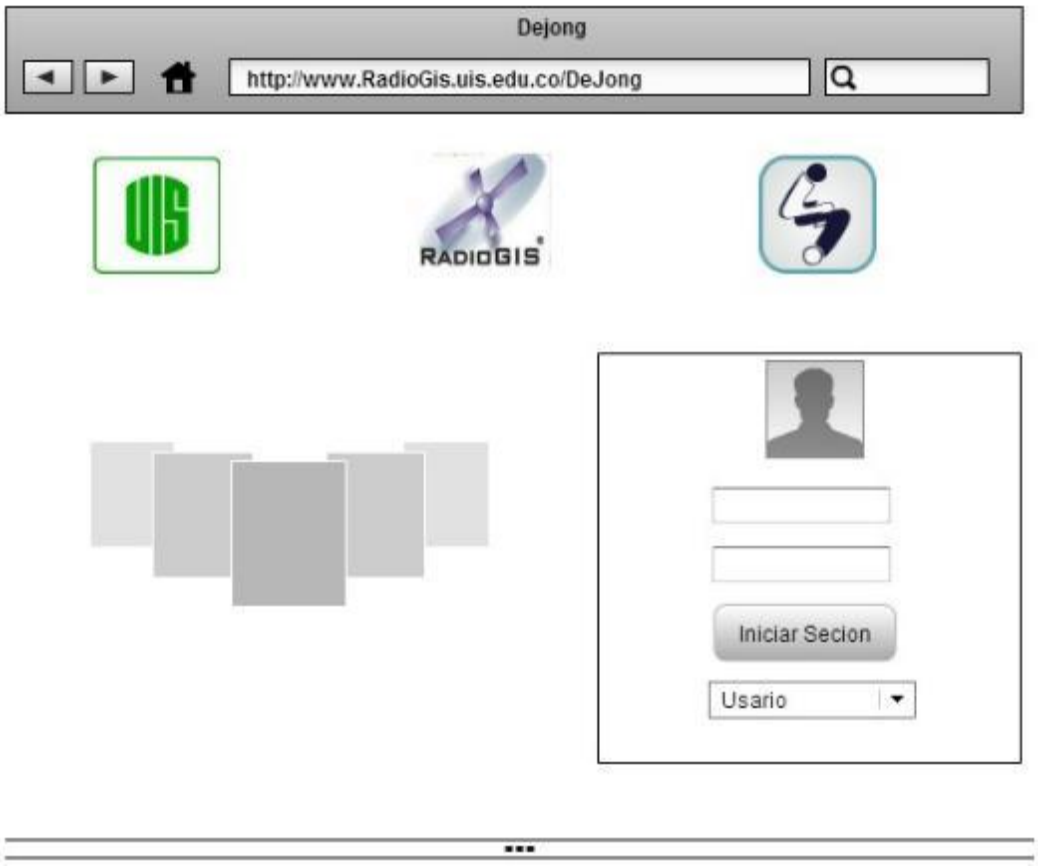
TRENDS IN 2015 : The Platform is King of the IoT Era,” 2015.

TRENDS T., ACCELERATE T., and PRODUCTIVITY Y., “NI Trend Watch 2015 Technology

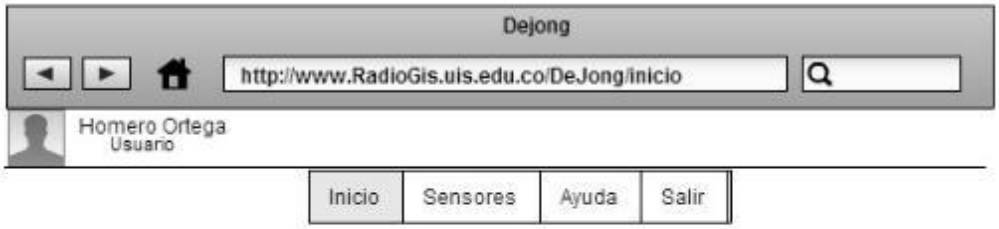
TRYGVE/MVC. [Online]. Available: http://heim.ifi.uio.no/~trygver/themes/mvc/mvc-index.html. [Accessed: 12-Sep-2015].

ANEXOS

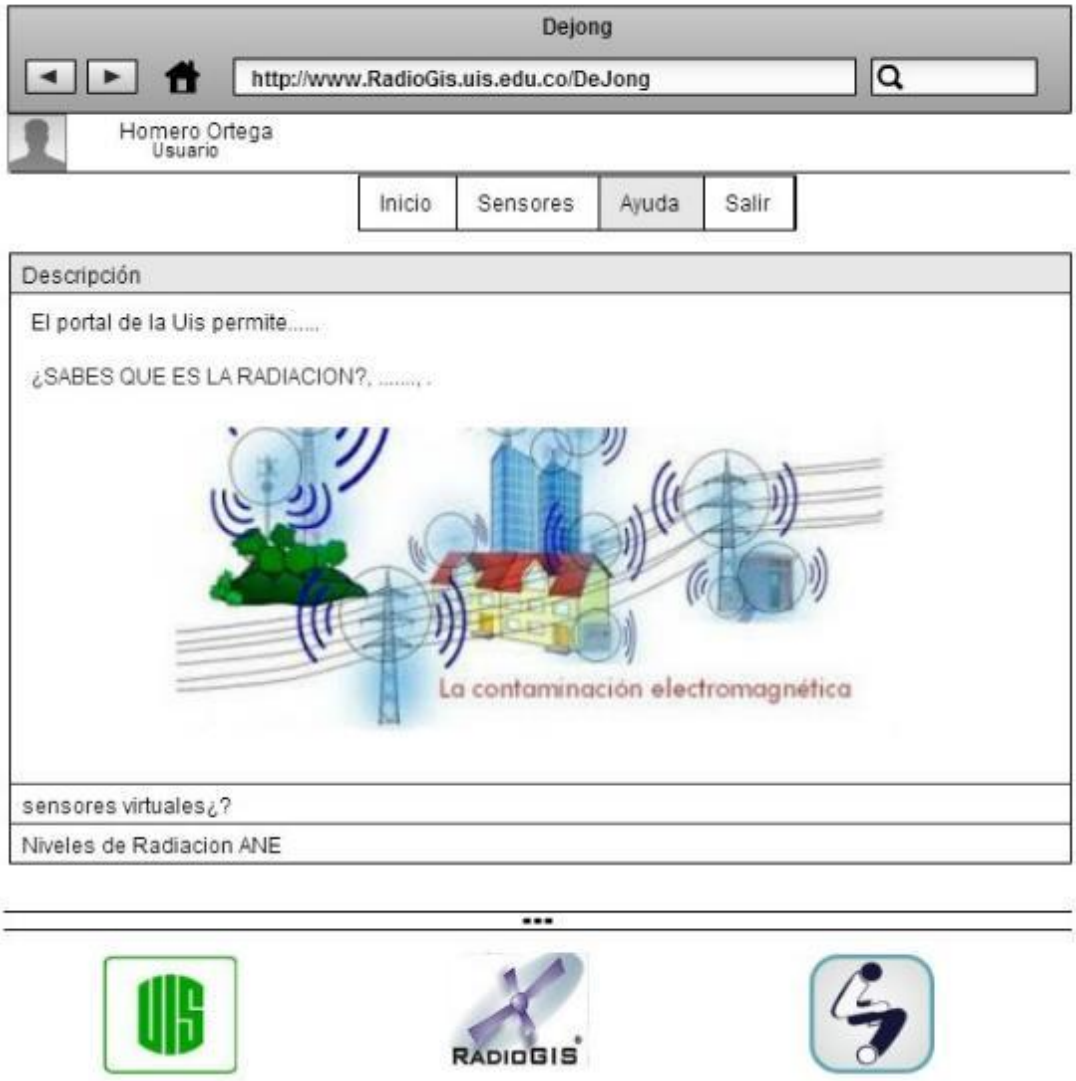
Anexo A: Mockups de la plataforma



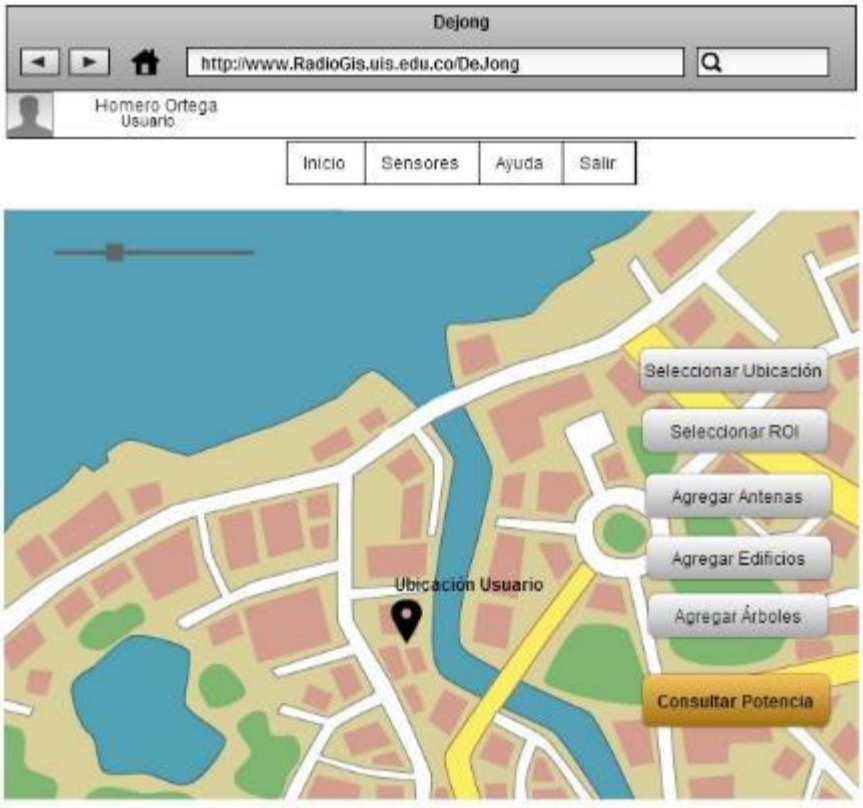

Dejong
http://www.RadioGis.uis.edu.co/DeJong/registro
USER
Administrador
Inicio
Sensores
Ayuda
Configuración
Salir
Formulario
Nombre
Apellido
Nombre de usuario
Contraseña
Confirmar clave
Correo Electrónico
Actividad (opcional)
Idioma
Español
Por favor rellene todos los campos. Esto le permitirá acceder a la apliación.
Registro
Cancelar



DESCRIPCION DE LA HERRAMIENTA


Dejong
http://www.RadioGis.uis.edu.co/DeJong
Homero Ortega
Usuario
Inicio
Sensores
Ayuda
Salir
Descripción
El portal de la Uis permite.....
¿SABES QUE ES LA RADIACION?, ..., .
La contaminación electromagnética
sensores virtuales¿?
Niveles de Radiacion ANE


Dejong
http://www.RadioGis.uis.edu.co/DeJong
Homero Ortega
Usuario
Inicio
Sensor
Ayuda
Salir
Seleccionar Sensores
DeJong-RadioGis
RADIOGIS


Dejong
http://www.RadioGis.uis.edu.co/DeJong
Homero Ortega
Usuario
Inicio
Sensores
Ayuda
Salir
Seleccionar Ubicación
Seleccionar ROI
Agregar Antenas
Agregar Edificios
Agregar Árboles
Consultar Potencia
Ubicación Usuario

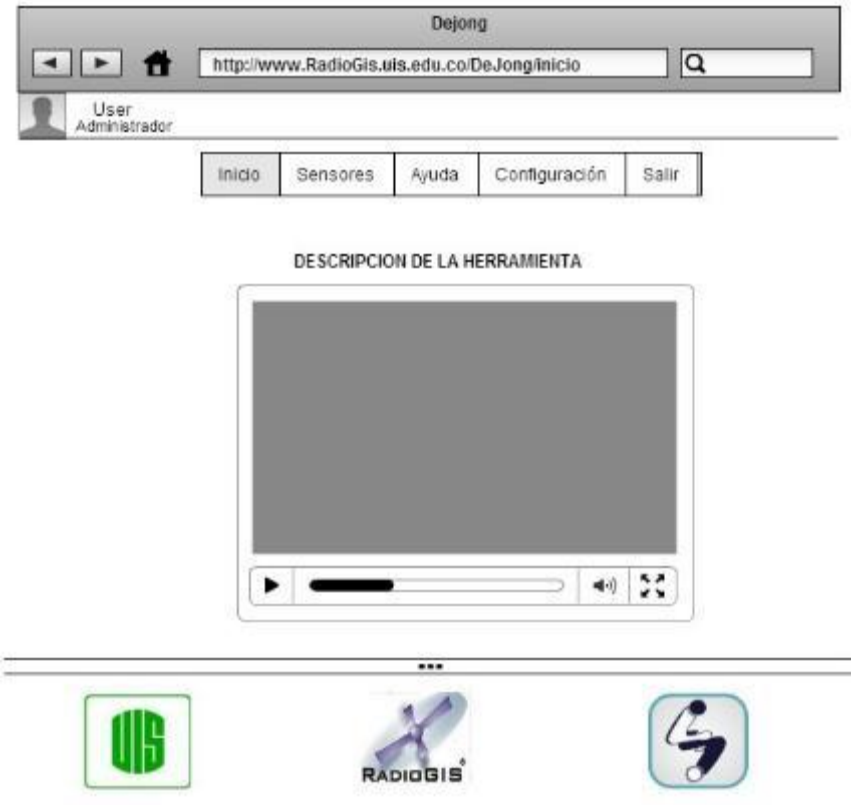

Dejong
http://www.RadioGis.uis.edu.co/DeJong
Homero Ortega
Usuario
Inicio
Sensores
Ayuda
Salir
Seleccionar Ubicación
Seleccionar ROI
Agregar Antenas
Agregar Edificios
Agregar Árboles
Consultar Potencia
ROI (seleccionado por el usuario)
Ubicación Usuario

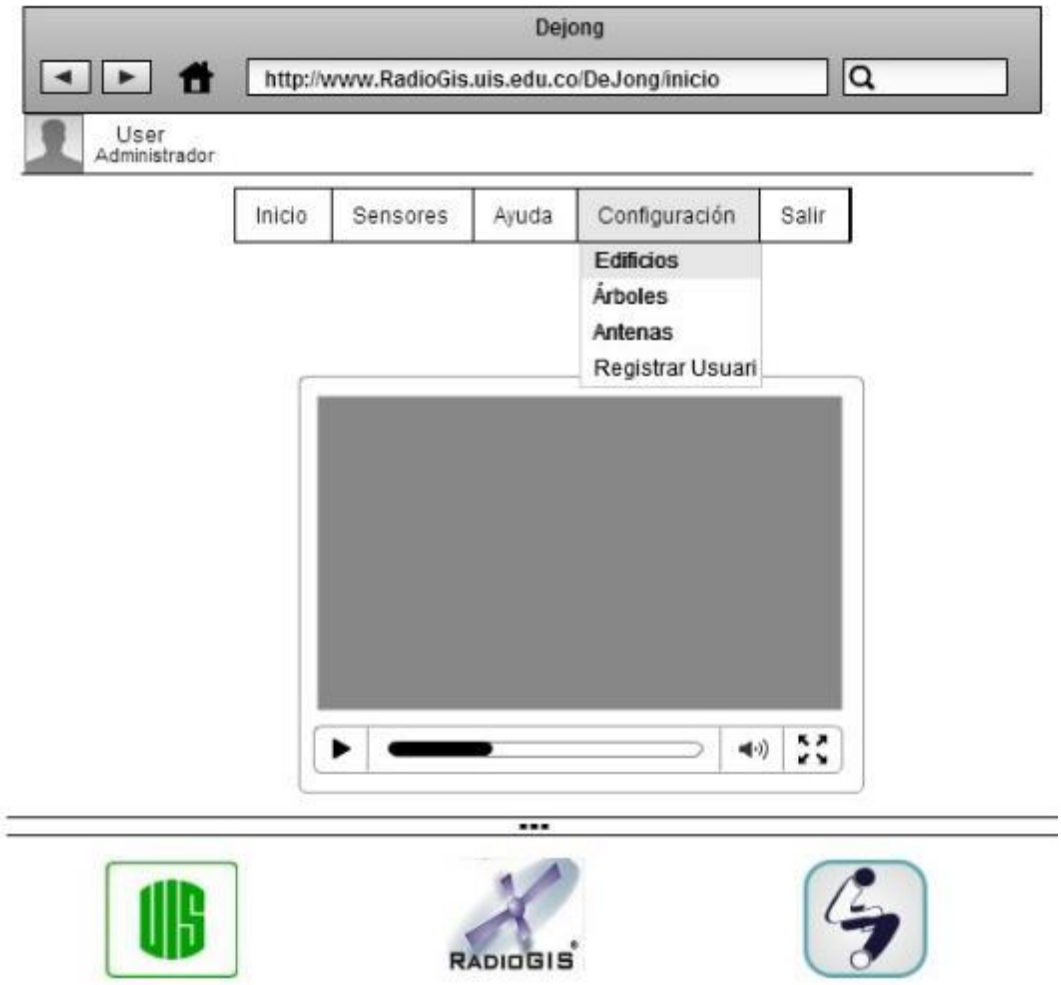

Dejong
http://www.RadioGis.uis.edu.co/DeJong
Homero Ortega
Usuario
Inicio
Sensores
Ayuda
Salir
Niveles de Exposición
Máximo permitido por la UIT: 100
0 - 25
26 - 50
51 - 75
76 - 100
101 - 120
ROI (seleccionado por el usuario)
Ubicación Usuario
Seleccionar Ubicación
Seleccionar ROI
Agregar Antenas
Agregar Edificios
Agregar Árboles
Consultar Potencia

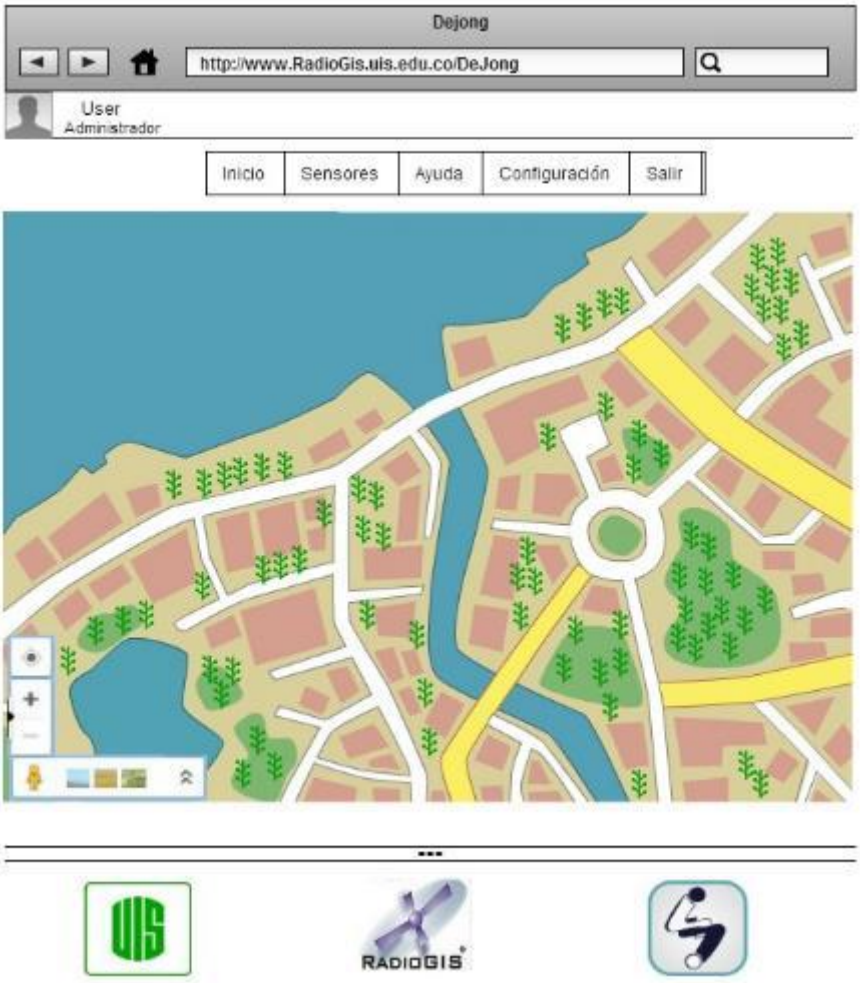

Dejong
http://www.RadioGis.uis.edu.co/DeJong
Homero Ortega
Usuario
Inicio
Sensores
Ayuda
Salir
Niveles de Exposición
Máximo permitido por la UIT: 100
0 - 25
26 - 50
51 - 75
76 - 100
101 - 120
ROI (seleccionado por el usuario)
Ubicación Usuario
Seleccionar Ubicación
Seleccionar ROI
Agregar Antenas
Agregar Edificios
Agregar Árboles
Consultar Potencia

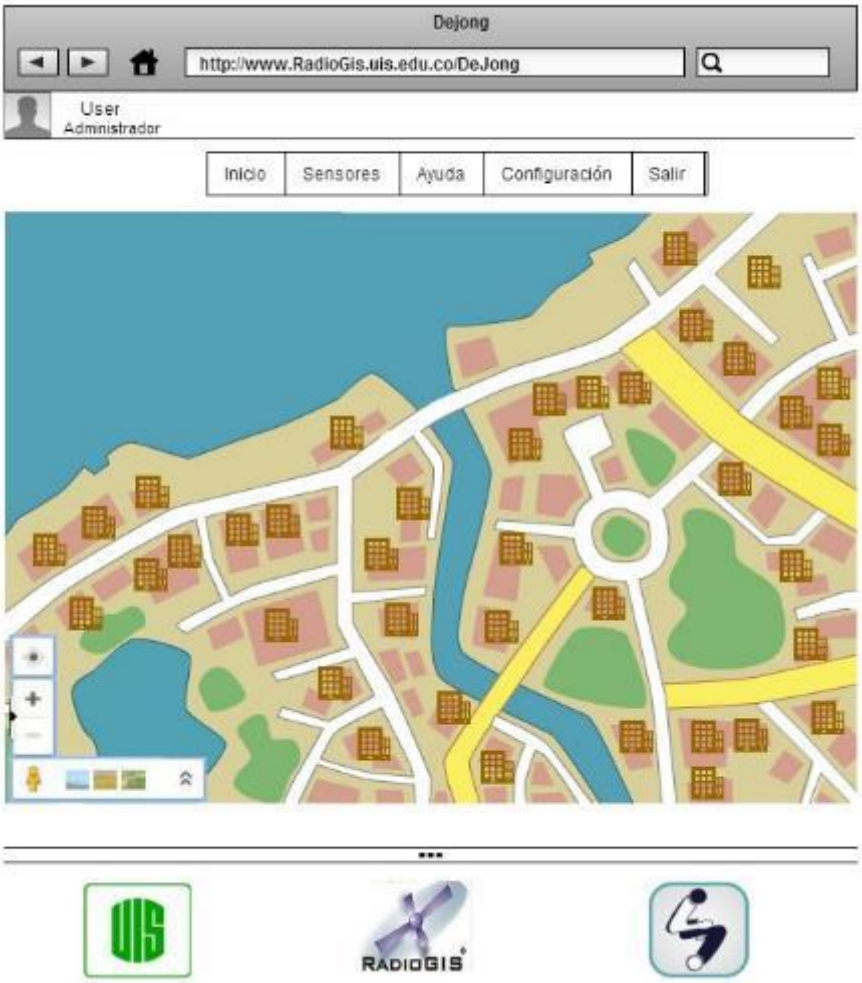

Dejong
http://www.RadioGis.uis.edu.co/DeJong
Homero Ortega
Usuario
Inicio
Sensores
Ayuda
Salir
Niveles de Exposición
Máximo permitido por la UIT: 100
0 - 25
25 - 50
51 - 75
76 - 100
101 - 120
ROI (seleccionado por el usuario)
Ubicación Usuario
Seleccionar Ubicación
Seleccionar ROI
Agregar Antenas
Agregar Edificios
Agregar Árboles
Consultar Potencia
RADIOGIS

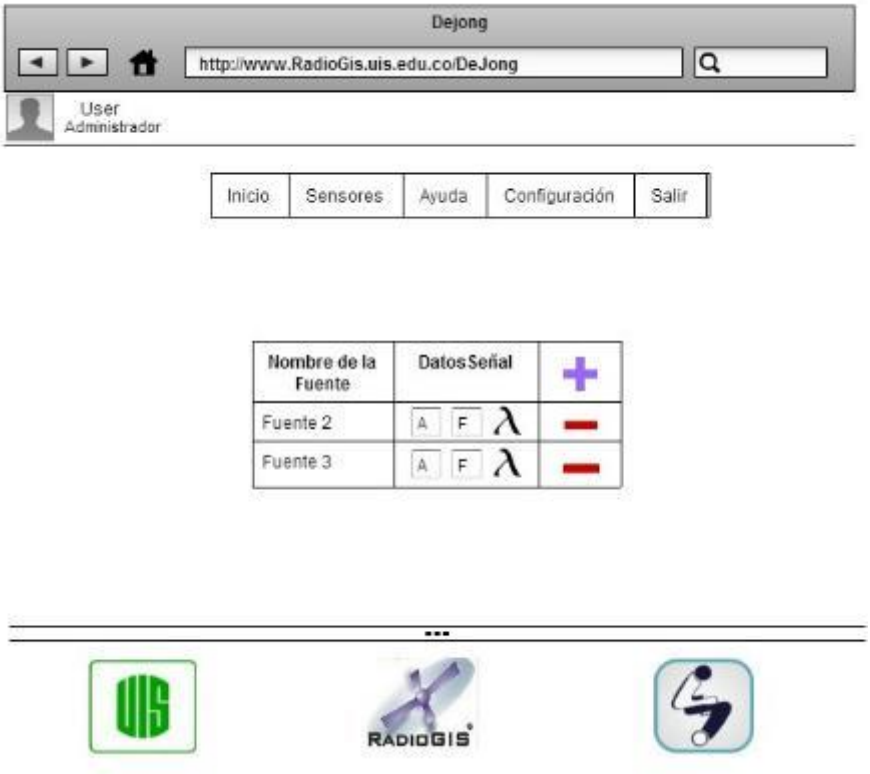

Dejong
http://www.RadioGis.uis.edu.co/DeJong
Homero Ortega
Usuario
Inicio
Sensores
Ayuda
Salir
Niveles de Exposición
Máximo permitido por la UIT: 100
0 - 25
26 - 50
51 - 75
76 - 100
101 - 120
ROI (seleccionado por el usuario)
Ubicación Usuario
Seleccionar Ubicación
Seleccionar ROI
Agregar Antenas
Agregar Edificios
Agregar Árboles
Consultar Potencia
RADIOGIS


Dejong
http://www.RadioGis.uis.edu.co/DeJong/inicio
User
Administrador
Inicio
Sensores
Ayuda
Configuración
Salir
DESCRIPCION DE LA HERRAMIENTA
RADIOGIS


Dejong
http://www.RadioGis.uis.edu.co/DeJong/inicio
User
Administrador
Inicio
Sensores
Ayuda
Configuración
Salir
Edificios
Árboles
Antenas
Registrar Usuari
RADIOGIS


Dejong
http://www.RadioGis.uis.edu.co/DeJong
User
Administrador
Inicio
Sensores
Ayuda
Configuración
Salir
RADIOGIS


Dejong
http://www.RadioGis.uis.edu.co/DeJong
User
Administrador
Inicio
Sensores
Ayuda
Configuración
Salir
RADIOGIS


Dejong
http://www.RadioGis.uis.edu.co/DeJong
User
Administrador
Inicio
Sensores
Ayuda
Configuración
Salir
Nombre de la Fuente
Datos Señal
Fuente 2
A
F
Fuente 3
A
F

Anexo B. Artículo publicado en la IEEEXplorer, sobre el trabajo realizado en este proyecto.

Link del documento:

http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7152102

IEEE COLCOM 2015

Implementation of DeJong radio propagation algorithm as a virtual sensor for cloud services to enhance radio spectrum management

Homero Ortega Boada
Head and researcher of RadioGis Group.
Professor at Universidad Industrial de Santander
Bucaramanga, Colombia
homero.ortega@radiogis.uis.edu.co

Manuel Guillermo Flórez
Head and researcher of Conuss Group.
Professor at Universidad Industrial de Santander
Bucaramanga, Colombia
mgflorez@radiogis.uis.edu.co

Juan David Rodriguez Ariza
Researcher of RadioGis Group
Engineer at Universidad Industrial de Santander
Bucaramanga, Colombia
jrodriguez@radiogis.uis.edu.co

Maria Alejandra Alfonso Muñoz
Researcher of RadioGis Group
Student at Universidad Industrial de Santander
Bucaramanga, Colombia
alejandra.alfonso@radiogis.uis.edu.co

César Augusto Pico Martinez
Researcher of Conuss Group.
Student at Universidad Industrial de Santander
Bucaramanga, Colombia
Cesar.pico@radiogis.uis.edu.co

Celso Andrés Forero
Researcher of RadioGis Group
Master (c) at Universidad Industrial de Santander
Bucaramanga, Colombia
Celso.forero@radiogis.uis.edu.co

***Abstract*— this paper proposes a strategy to implement radio propagation algorithms as cloud services using cloud computing to support radio spectrum management. The DeJong radio propagation algorithm was selected to be implemented. The strategy supports not only radio propagation algorithms but also a combination of hardware measurement and software simulation as well as other applications where sensors are used. The strategy involves mainly virtual sensors which can run in virtual servers such as web services and therefore as part of Internet of Things (IoT) technology. Many benefits arise from this solution and it offers possibilities for different kind of specialists and users who can cooperate for the development of services in a flexibly way, as system engineers are freed of the heavy task of sensor implementation and are encouraged to focus in architecture scalability, security and exploring high availability of services. The mere fact of having implemented the DeJong algorithm as a cloud service represents an important contribution to Spectrum Management technics, especially for those looking forward to Advanced IMT technologies where other algorithms are not effective.**

Keywords — De Jong Algorithm; virtual sensors, cloud computing, service architecture, spectrum management.

I. INTRODUCTION

Radio Spectrum (RS) Management Systems are increasingly gaining amazing capabilities. Nowadays, these systems not only include spectrum databases and algorithms to support the allocation process of frequency bands, but also involve modern technology to make measurements that reveals the actual occupation of the RS. However, these kind of solutions are being developed as proprietary and closed systems which hinders the participation of experts in the fast implementation of solutions to the increasing number of challenges – this fact goes in the opposite direction of the global trends. Radio propagation simulators, which may be useful for several purposes, come into play along with the use of modern RS measuring equipment, such as spectrum analyzers or non-ionizing radiation (NIR) measure instruments. For example, when some measurements are compared with simulation results in the same conditions, it is possible to identify deviations large enough so as to identify if they are due to spectrum infringement or spectrum misuse. The simulations also provide a low cost possibility to know what would happen if certain communications technologies are implemented in certain geographic areas. For instance, when a user with a low-cost and low capacity mobile terminal opens an application, he can see how non-ionizing radiation (NIR) levels varies as he moves, since his terminal is an NIR meter. When using radio propagation algorithms in the cloud, it is possible to offer these low-cost services to users without the requirement for expensive equipment to make measurements.

This paper presents the breakthrough innovation of considering the potential of this propagation algorithm as a virtual device that provides spectrum measurement information.

Radio propagation algorithms may be used as real sensors as well as measuring equipment used in spectrum monitoring. Both of which can be associated with the concept of the Internet of Things (IoT), broadening the benefits of modern

978-1-4799-8834-1/15/$31.00 ©2015 IEEE

technologies. Even the technology research firm Gartner estimates that soon there will be more devices connected to the Internet than humans on the planet. Moreover, by 2020 each house could have more than 500 devices connected [3].

The problem to be solved is the need to answer the following question: How can a radio propagation algorithm, with high computing demand as the DeJong algorithm, be considered a virtual sensor that runs in the cloud to provide services to multiple users who may have low performance terminals?

Cloud computing is changing many paradigms. For instance, an architect could rely on cloud computing only to recalculate any building´s parameters with only the help of his cellphone. Currently, a wide range of intelligent and costly terminals dominates the market. It is estimated that by 2020 most of the terminals will be those of low computing performance and consequently inexpensive. Not only virtual sensors and the proposed implementation methodology will have undoubtedly a major impact on the way spectrum management systems are developed in bringing services to people, but this will also be valuable to all type of applications where simulators are used for predicting the behavior of any kind of variables in fields like agriculture, industry, environment, mineral exploration, medical care, etc. This paper also draws the road to engineers of various professions who need to complement each other in an efficient way to develop the IoT services. This paper is the result of some projects developed at the Industrial University of Santander in cooperation with RadioGis and Conuss groups, including the project "Desarrollo de interfaz software para desplegar modelos de simulación en la Internet y aplicación web para simular el modelo de radio propagación de De-Jong" and other projects aimed to an IoT platform development.

II. THEORETICAL FRAMEWORK

A. *The cloud and web services*

With significant advances in Information and Communications Technologies (ICT) in the last half century, there is a vision that computing will one day be the fifth utility (after water, electricity, gas and telephone). The computing, like all the other four existing utilities, provides the basic level of a service that is considered essential to meet the everyday needs of the community in general [4].
This is why, since the late 1990s is talked about the service-oriented computing (SOC), as an important paradigm that is changing the way that software applications are designed, how they are delivered and how they are consumed. Although the structure of web services has been heavily researched, it is yet to be completed specially when it comes to taking full advantage of the many advances such as cloud computing, social networks and the Internet of Things [5].

The Cloud Computing is an internet-based computer system, which uses virtual machines and remote data centers in order to manage information services and file management applications. This grants applications to run without having to be installed on computers, allowing it to become a new technological paradigm of great social impact [6].

There are extensive benefits to cloud computing some are: most of the software does not require any additional application, it is a fast infrastructure that allows updates constantly, it is adaptable to large changes, it allows access to most systems anywhere is found, it is low-priced given that it does not require too much hardware [6].

The Cloud is leading a new industrial revolution supported in Data Centers and factories of «Web Applications» (Web Apps). This new revolution will produce a great social, economical and technological change.

Web services are software systems that allow data and functionality sharing between applications over a network. This is supported by different standards that ensure interoperability of services. Various software applications are developed in different programming languages, and executed on any platform. All of those can use web services to exchange data in computer networks like the Internet [4].

B. *Model View Controler (MVC)*

Nowadays, anywhere in the world, building applications are focused on two aspects, firstly, how to build better applications faster, and secondly, how to use a greater number of standards in application design that would optimize the code usage and better maintenance of the systems developed [2].

The MVC can be considered as a pattern design of a visual software architecture that divides any software system based on the representation of information, data model and application control.
The model consists of data application and business rules, the "view" mode can present information to the user, can be any output representing the data; the controller is responsible for responding to events (typically user actions) and invoke requests to the 'model' when a request for information is made, You can also send commands to your 'view' associated if a change is requested in the way they should present information.
We could say that the controller is the brain of the service, not the smartest part, but it serves cleverly the user to then invoke the necessary tools in the model, to meet needs of data query, data processing or remote control equipment, or in the view, to meet needs of visualization of the results. We could imagine the user in the "view" layer that is closest to the user, but in logical terms is the driver who attends the user.

Although you can find different implementations of MVC, the control flow that follows is usually presented in *Fig.1*. The user interacts with the view of some form (for example, the user presses a button).

1) The controller accesses the model, updating, possibly modifying it appropriately to the action requested by the user (for example, the controller updates the product of a purchase to a user).
2) The model delivers the request data to the controller
3) The controller delivers you to view model data to generate the appropriate user interface where changes in the model (eg reflected produces a listing of the contents of the product in a purchase).
4) The view controller and the delivery is delivered to the user interface, which expects new user interactions, beginning the cycle again.

The MVC pattern has several advantages among which is the clear separation between the components of a program, which allows separate implementation also features an Application Programming Interface API (Application Programming Interface) well defined providing the opportunity for anyone using the API, can replace the Model, View and Controller, without apparent difficulty [2].

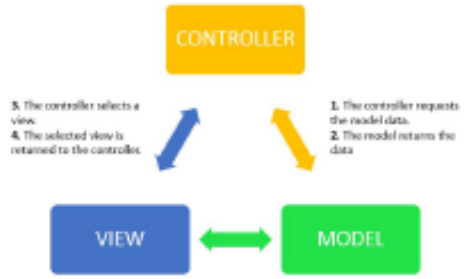


Fig. 1. View Control Model.

C. virtual Sensor

Monitoring systems often use spectrum measuring equipment from the viewpoint of services equals sensors. Although the DeJong's propagation algorithm is only software, also provides similar information measuring equipment is used as a virtual simulations service that enables access algorithm and simulations of measurements, which could play a key role in controlling and planning UHF networks.
Here started to talk about virtual sensors. Very few works speak of virtual sensors, also known as "soft sensors", these sensors are algorithms using measurements from real sensors or data from other sources, which could be used to predicting behavior of situations that require considerable planning or prevention level in execution.
The use of virtual sensors allows the simulation of non-ionizing radiation measurements (NIR), with detailed information from antennas and geography in a region of interest (ROI), reducing the cost of measurement services, because it does not depend on physical computers.
In other cases the virtual sensors could be a necessary complement to any type of problem, as in the case of a doctor who considered insufficient the information provided for an electrocardiogram and requires more information and tools for processing.
The term virtual sensor is new, it is defined as the association of traditional sensors (hardware) with an estimation algorithm (software) to provide estimates of unmeasured variables through the network, with variable measurement dead time, or parameter model [8].
Currently, the virtual sensors are used for example to monitor oil pipelines and also for simulation of medical patients to anticipate the behavior of a disease [9].

D. The De Jong Algorithm

In 1997 Dr. Leon Yvo De Jong began radio propagation experiments in various urban environments in Europe to find out the main causes of imprecisions commercial simulation programs have to predict the coverage provided by new cellular micro cells networks. His concern was the type of cells that are formed when low power antennas are mostly installed below the roofs of buildings. This is of great importance for the fourth generation (4G) mobile communications but also for the fifth generation (5G). The entry of the 5G mobile communications will be accompanied of more a more low-power and little discernible antennas installed almost in each building of a city just below their roofs. In these circumstances, as it was proven by De Jong, traditional algorithms used to simulate the coverage of mobile networks do not work. It is necessary to consider many of the phenomena that arise from wave propagation and their interaction with the various obstacles of a city. That's why De Jong algorithm is highly computational demanding. The following steps proposed by the RadioGis group as part of an original computational model to implement the algorithm can provide an overview to better understand it [1]:

- The user draws the limits for the Region of Interest (RoI), ie he selects a geographic area in which he wants to observe the signal coverage from one or more antennas
- The user can also set the location of the antennas and their parameters
- The user can select a Point of Interest (PoI) or all RoI for the observation
- A geo referencing algorithm translates all the buildings of the RoI into geometric figures. It is necessary even when the user is interested just in a single point, because signal contributions reflected from all buildings may be high enough to change results
- An algorithm for virtual sources identification determines the walls of the buildings that may be considered as mirrors. It is the similar result of turning on the lights of a house with many mirrors inside. Each mirror reproduces the bulbs creating as many virtual sources as the number of neighboring mirrors and the bulbs the house has. That's why the term "virtual source" is used and some parameter like power, beam shape (similar to the radiation pattern of an antenna), phase shift and other have to be defined

- But De Jong algorithm goes farer in the identification of virtual sources. Thus, the trees cause waves scattering and therefore are considered by the algorithm as new special virtual sources. The same thing applies to the corners of buildings which cause waves diffraction.
- Higher order virtual sources may be identified when a ray of a virtual source reaches a mirror
- The virtual sources algorithm have to be configured properly so that the order of the virtual sources could have a limit, ie only most significant sources are taken into account.
- The next step is performed by a traditional simple algorithm that calculates in line of sight the contribution of every virtual source to a PoI o a RoI.
- The final step is the presentation of information to the user.

In the *Fig. 2* the external box corresponds to the RoI, the rectangles represents the buildings, the mirrors are the walls of such buildings in color different to black, the BS is the base station or the real source, the other stars are the virtual sources calculated by the algorithm and the colored areas represent the coverage that each of these sources provides.

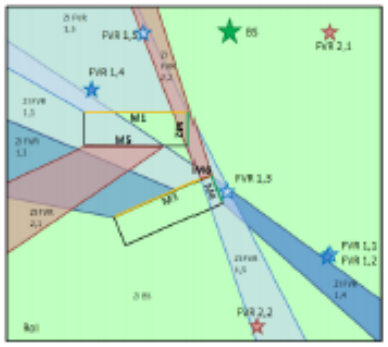


Fig.2. De Jong virtual sources

Thus, the De Jong algorithm is inspired in Huygens principle, so that every obstacle is translated to a virtual source, giving relevance to line of sight fading but also to walls crossing of the waves, to waves reflection from the buildings walls, to wave diffraction from building corners as well as to scattering by the trees [1]. The algorithm does not take into account the heights nor the geographical relief, because it is not justified when the micro wave sources are of low power, but the algorithm is opened to any possibility. A version for Colombia is proposed in the article "Modified Version of De Jong Radio propagation Model" [1], where the scattering phenomenon, that occurs when a wave collides with tree branches, is treated as a new kind of virtual dispersion sources.

In this paper, the De Jong algorithm has been implemented as a virtual sensor in its entirety without the use of specialized hardware, which allows the development of interesting applications in a very flexible way.

III. SERVICE PLATFORM BASED ON VIRTUAL SENSOR

Simulations are a perfect complement to telemetry and consequently to support the management of radio spectrum. This raises the need to integrate simulations service platforms in which converge electronics and computing. To achieve this it is necessary first to review the most suitable for this type of architecture implementations taking into account national and international betting IIoT and the benefits it can have for promoting the development of services which converge the most diverse professions and technologies. In a IIoT architecture measuring equipment can be treated as sensors that provide georeferenced information to a processing system in the cloud. Precisely radio propagation algorithms also provide results, but are simulated, correspond to georeferenced measurements. This is where the need to implement the algorithm Dejong as a virtual sensor on a platform supporting IoT management ERE arises.

A. Virtual sensor platform.

In the proposed platform for this work is to show how they can or should live virtual sensors with real-looking as they should be supplemented. We rely on the MVC design pattern as explained below *Fig.2.*

1) Data layer: In this layer are measurement data proporsionados by real sensors or data from simulations by virtual sensors that help business logic, this layer is an application model. Having databases is always necessary, since otherwise the measurements and the results they deliver the models would be volatile.

2) Business logic: Is responsible layer logic and algorithms that simulates the behavior of a real sensor. What is interesting is that a programmer can see virtual sensors in the same way as real, because each of them, whether virtual or real is seen as a Web Service. This is the reason why in this layer have the real sensors also located.

3) Services layer: In this layer services that meet specific needs for users are configured. It is basically of applications that can take the form of webservices or not but they serve the user, using the resources available in the business logic and data layer, and incluoso elegiemdo views that best suit the user.

4) View: It is the layer where end users access are the interfaces of the platform can be mobile applications, third party applications, drivers. With this vision a user terminal could be as simple as to support the view, while the bulk of the application running in the cloud.

To demonstrate the difference in the processing of data compared to a virtual sensor real sensor, take as an example the possibility that there is a "drone" that is installed to a sensor measuring the charge of non-ionizing radiation (NIR). From the viewpoint of the software platform, the meter located in the drone is seen as a sensor as a Web Service actual measurement. The user has an application that is nothing but a view of the service to order the opening and recording of measurements, the order is received by a service measurements considered the controller and proceeds to consume the Web Service from actual measurement finally deposited the measurement data in the corresponding database in the cloud. For visualization, the user enters a new view of the user interface that communicates with a new service or hosted in the cloud controller. This selects the webservice necessary to present the information needed by the user

Now we will provide a similar description using a virtual sensor. First, the user displays a view through any terminal, and service requests sensor simulation, then this service is connected with the business logic that has Dejong algorithm, which is responsible for processing and return an answer, if the user desires. We realize that for the type of platform proposed real sensors have the same treatment as virtual sensors.

Fig.3. Virtual sensor platform.

B. Development of virtual sensors

A virtual sensor can be viewed as a simulation of one or more real sensors such as the sensor which is to combine the strengths of a real sensor with an algorithm or simply as the result of combining multiple sensors of different natures. The advantage of using this concept is the possibility to define a particular area of development for a group of experts in sensors, usually electronic engineers, in our case, experts in radio. They handle constantly improve sensing techniques at both the hardware and software without having to worry about what happens in the cloud. Rather focus their abilities and right sensor development resources, be they real or virtual. Another group of experts in services specializing in the development of end applications including the view and the controller without having to worry about external hardware or complex digital processing algorithms. Finally you can have a group of experts in architecture, usually systems engineers, who are engaged in maintaining and improving the efficiency of the platform and its services. This convergence of human and technical resources for the development of IoT which in turn represents a convergence between electronics and computers is achieved.

The implementation made in the course of this project for the virtual sensor corresponding to the algorithm De Jong, presents an interesting alternative for the group of experts in sensors. What was done was create a virtual machine in the cloud dedicated solely to De Jong algorithm. In turn, the algorithm De Jong was supplemented so it could be viewed as a Web Service. As a result, the panel sensor could continue to participate in improving the algorithm, working on the virtual machine in the same way as they would on any computer. Or you could work on your computer to finally copy the virtual machine simulation software developed. Meanwhile, the expert group services could work on developing a service that consumes the web service for the algorithm De Jong and a web service that is used to store the census of any sensor either virtual or real.

Separate multiple Virtual Machines also useful as part of other activities such as performing maintenance (upgrades and backups) efficiently, to implement security strategies and an efficient measure for high availability. For example, the data layer stayed in a machine with Windows Server and the business layer and service on a machine running Linux; Queries are handled including via secure requests for database queries. This model has several advantages such as stability since being divided into different virtual machines each service is independent, eg the possible collapse of one of the machines do not fall platform for other services; security; accessibility; high performance. We also have some disadvantages such as: dependence on internet, limited to the speed of the connection.

IV. RESULTS

Based on the Java implementation developed by Melgarejo [11], in which an algorithm that simulates radiation from many virtual sources for UHF wireless network planning is done, the algorithm is parameterized to migrate from a desktop application to a web service, using REST-FULL [12] web services. The web service is designed to receive and respond requests in JSON [13] format, the serviced is complemented with additional Java methods for converting geographic coordinates to a particular format that this algorithm handles.

The graphical environment is created using HTML and Google Maps technology as two basic services: the first one is to support the user in the algorithm initial configuration including the definition of a RoI, edition of buildings, location

of trees and a location for the virtual sensor which corresponds to the geographical point at which the user wishes to perform a virtual measurement. AJAX [14] (a dynamic web method) is used to take important data points on the map and with them construct the JSON request and sent to the server using the POST method [15], for safety and given the amount of data required for simulation; the second service invokes the virtual machine in the cloud that makes the algorithm computational process, this communication is achieved through web service that receives the above mentioned coordinates and then sends the response with a power value corresponding to the point in which is located the virtual sensor.

The *Fig 4* shows how the user achieves modify the RoI coordinates and position one or more radio bases.

Fig. 4. Prototype simulation.

V. FINDINGS

The virtual and real sensors have the same behavior in the proposed model to the user, facilitating the design and implementation of complex services

The feasibility and efficiency of architecture with processing on the cloud is demonstrated with the complex Dejong algorithm, converting a desktop application into a service you get from a few benefits terminal with internet access algorithm.

The use of simulation services that are provided by the DeJong virtual sensor could support certain management tasks ERE as create applications that convert the user's cellphone on a power levels meter or even RNI; making considerations about what happened, if you add or if one or more base stations are removed from certain sites; and compare measurements with simulations to identify clandestine broadcasts, etc.

Virtual sensors measurement of the ERE are key to predict measurements in varying conditions such as terrain, the entry of new technologies, antennas configuration, etc.

REFERENCES

[1] M. D. Pineda Albucema, H. Ortega Boada, and A. Navarro Cadavid, "Modified Version of de Jong Radio Propagation Model," *Int. Informatics Telecommun. Symp. I2TS2010*, pp. 141–147, 2010.

[2] P. Modelo-vista-controlador, "Patrón Modelo-Vista-Controlador.," vol. 11, no. 1, pp. 47–57, 2012.

[3] T. Trends, T. Accelerate, and Y. Productivity, "NI Trend Watch 2015 Technology Trends in 2015□: The Platform is King of the IoT Era," 2015.

[4] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Futur. Gener. Comput. Syst.*, vol. 25, no. 6, pp. 599–616, Jun. 2009.

[5] Q. Z. Sheng, X. Qiao, A. V. Vasilakos, C. Szabo, S. Bourne, and X. Xu, "Web services composition: A decade's overview," *Inf. Sci. (Ny).*, vol. 280, pp. 218–238, 2014.

[6] "Computacion en Nube." [Online]. Available: http://www.computacionenmbe.org/computacion-en-nube/. [Accessed: 03-Mar-2015].

[7] L. Qian, Z. Luo, Y. Du, and L. Guo, "Cloud computing: An overview," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 5931 LNCS, pp. 626–631, 2009.

[8] J. C. B. Gonzaga, L. a C. Meleiro, C. Kiang, and R. Maciel Filho, "ANN-based soft-sensor for real-time process monitoring and control of an industrial polymerization process," *Comput. Chem. Eng.*, vol. 33, pp. 43–49, 2009.

[9] B. M. Lee and U. G. Kang, "Virtual Sensor for Diabetes Meter in U-Health Service," vol. 6, no. 6, pp. 87–96, 2014.

[10] J. Wei, Y. Xu, and J. Zhang, "Neural networks based model predictive control of an industrial polypropylene process," in *Proceedings of the International Conference on Control Applications*, vol. 1, pp. 397–402.

[11] Y. H. M. Melgarejo, H. O. Boada, and M. D. P. Albucema, "A model for urban microcell radio propagation prediction focused on reliable

implementation," *2012 IEEE Colomb. Commun. Conf. COLCOM 2012 - Conf. Proc.*, no. 1, 2012.

[12] "What Are RESTful Web Services? - The Java EE 6 Tutorial." .

[13] "Introducing JSON." .

[14] "¿Qué es AJAX? | Digital Learning - Formación online en Nuevas Tecnologías." .

[15] "What is GET and POST method in HTTP and HTTPS Protocol." .