UNIVERSIDAD POLITÉCNICA SALESIANA

SEDE QUITO

CARRERA DE COMPUTACIÓN

DESARROLLO DE UNA APLICACIÓN DE STREAMING PARA EL ENVÍO DE IMÁGENES METEOROLÓGICAS DESDE UN RADAR FURUNO MANEJADO POR EL INAMHI.

Trabajo de titulación previo a la obtención del

Título de Ingeniero en ciencias de la Computación

AUTORES:

KEVIN ANDRÉS CHANGOLUISA CUAYAL

BRYAN ANDRÉS IMBAQUINGO ALMAGRO

TUTOR:

PAULINA ADRIANA MORILLO ALCÍVAR

Quito-Ecuador

2022

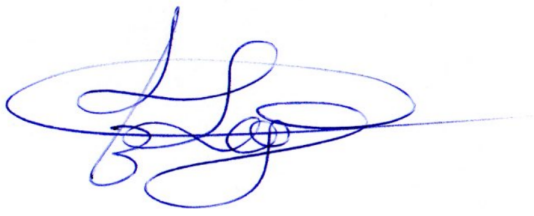
CERTIFICADO DE RESPONSABILIDAD Y AUTORÍA

DEL TRABAJO DE TITULACIÓN

Nosotros, Kevin Andrés Changoluisa Cuayal con documento de identificación No. 1718123563, y Bryan Andrés Imbaquingo Almagro con documento de identificación No. 1720457330, manifestamos que:

Somos los autores y responsables del presente trabajo; y autorizamos a que sin fines de lucro la Universidad Politécnica Salesiana pueda usar, difundir, reproducir o publicar de manera total o parcial el presente trabajo de titulación.

Quito, 11 de marzo del año 2022



.....

Kevin Andrés Changoluisa Cuayal

1718123563

.....

Bryan Andrés Imbaquingo Almagro

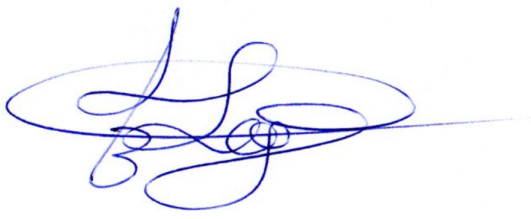
1720457330

CERTIFICADO DE CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE TITULACIÓN A LA UNIVERSIDAD POLITÉCNICA SALESIANA

Nosotros, Kevin Andrés Changoluisa Cuayal con documento de identificación No. 1718123563, y Bryan Andrés Imbaquingo Almagro con documento de identificación No. 1720457330, expresamos nuestra voluntad y por medio del presente documento cedemos a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que somos autores del Proyecto Técnico: ”Desarrollo de una aplicación de streaming para el envío de imágenes meteorológicas desde un radar furuno manejado por el Inamhi”, el cual ha sido desarrollado para optar por el título de: Ingeniero en ciencias de la Computación, en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En concordancia con lo manifestado, suscribimos este documento en el momento que hacemos entrega del trabajo final en formato digital en la Biblioteca de la Universidad Politécnica Salesiana.

Quito, 11 de marzo del año 2022



.....	.....
Kevin Andrés Changoluisa Cuayal	Bryan Andrés Imbaquingo Almagro
1718123563	1720457330

CERTIFICADO DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN

Yo, Paulina Adriana Morillo Alcívar con documento de identificación No. 1715646574, docente de la Universidad Politécnica Salesiana, declaro que bajo mi tutoría fue desarrollado el trabajo de titulación: DESARROLLO DE UNA APLICACIÓN DE STREAMING PARA EL ENVÍO DE IMÁGENES METEOROLÓGICAS DESDE UN RADAR FURUNO MANEJADO POR EL INAMHI, realizado por Kevin Andrés Changoluisa Cuayal con documento de identificación No. 1718123563 y por Bryan Andrés Imbaquingo Almagro con documento de identificación No. 1720457330, obteniendo como resultado final el trabajo de titulación bajo la opción Proyecto Técnico, que cumple con todos los requisitos determinados por la Universidad Politécnica Salesiana.

Quito, 11 de marzo del año 2022

Atentamente

.....

Ing. Paulina Adriana Morillo Alcívar, MsC

1715646574

DEDICATORIA

Dedico este proyecto de titulación a mi madre Felisa Honoria Cuayal, por el apoyo brindado en todas mis decisiones, por confiar en mi cuando decidí estudiar la carrera de Ingeniería en Ciencias de la Computación y por todo su esfuerzo para poder culminar mi carrera universitaria.

Kevin Andres Changoluisa Cuayal

Dedico este proyecto a mi familia en especial a mi madre Elvia Rocio Almagro Aguas, por ser la persona que me ha apoyado toda mi vida, ser una persona con la que se puede confiar y un modelo ejemplar a seguir, por todos los valores inculcados. Todo su esfuerzo hizo posible la culminación de mi carrera universitaria y mis hermanos por apoyarme en cada instante de mi vida.

Bryan Andrés Imbaquingo Almagro

AGRADECIMIENTOS

Queremos expresar nuestro agradecimiento a:

Nuestros docentes que fueron la guía en todo este camino profesional, en especial a nuestra tutora MsC. Paulina Adriana Morillo Alcivar, por su ayuda, paciencia y dedicación, por habernos guiado en este proyecto, en base a su experiencia y sabiduría, por esa pasión por su labor, una excelente persona y profesional.

A la Universidad Politécnica Salesiana que ha contribuido en nuestra formación académica y personal, forjando “Buenos cristianos y honrados ciudadanos”.

Finalmente agradecemos al Ingeniero Darwin Andrés Rosero Vaca, responsable de la Dirección de Información Hidrometeorológica del Instituto Nacional de Meteorología e Hidrología, por darnos la apertura para desarrollar este proyecto técnico, por su constante apoyo, sus indicaciones y orientaciones indispensables en el desarrollo de este proyecto.

KEVIN CHANGOLUISA **BRYAN IMBAQUINGO**

Índice General

Índice de Tablas

Índice de Figuras

RESUMEN

El presente proyecto técnico muestra el desarrollo de una aplicación web para el Instituto Nacional de Meteorología e Hidrología del Ecuador (INAMHI), la cual permite visualizar datos meteorológicos de la ciudad de Quito, transmitidos en tiempo real.

Los datos meteorológicos transmitidos contienen información previamente procesada de la intensidad de reflectividad receptada por el radar meteorológico Furuno. Esta información puede ser visualizada de manera gráfica, permitiendo conocer la concentración de las aguas lluvias con una mejor resolución en tiempo y cobertura.

La implementación incluye el procesamiento de datos del radar para extraer la información almacenada de archivos binarios procedentes de la lectura y la aplicación de técnicas de tratamiento de imagen en la información decodificada, para eliminar el eco parásito de la señal y mejorar la calidad de la imagen.

Para la transmisión de información entre el cliente y el servidor se implementó la tecnología de WebSocket. Este protocolo permite una comunicación bidireccional persistente y de baja latencia entre la aplicación web y el servidor web, facilitando la transferencia de datos en tiempo real.

Los resultados de la implementación muestran que la aplicación web es funcional y cumple con los requerimientos solicitados por el INHAMI. la aplicación permite el monitoreo gráfico de las zonas de la ciudad donde se están produciendo precipitaciones, mejorando las capacidades operativas de esta institución.

Palabras Clave: servidor FTP, *Backend*, angular, soketio, web socket, flask, procesamiento de imágenes, precipitaciones, hidrometeorológica, georeferencia.

ABSTRACT

This technical project shows the development of a web application for the National Institute of Meteorology and Hydrology of Ecuador (INAMHI) that allows the visualization of meteorological data of Quito, broadcast in real time.

The transmitted meteorological data contains previously processed information of the intensity of reflectivity received by the Furuno weather radar. This information can be displayed graphically, allowing to know the concentration of rainfall with a better resolution in time and coverage.

The implementation includes the processing of radar data to extract the information stored in binary files from the reading and the application of image processing techniques in the decoded information to eliminate the parasitic echo of the signal and improve the quality of the image.

For the transmission of information between the customer and the server, the WebSocket technology was implemented. This protocol allows a persistent and low latency bidirectional communication between the web application and the web server, facilitating the transfer of data in real time.

The results of the implementation show that the web application is functional and meets the requirements requested by INHAMI. The application allows monitoring graphically the areas of the city where rainfall is occurring, improving the operational capabilities of this institution.

Keywords: FTP server, *Backend*, angular, soketio, web socket, flask, image processing, rainfall, hydrometeorological, georeference.

INTRODUCCIÓN

Antecedentes

El Instituto Nacional de Meteorología e Hidrología (INAMHI) es la entidad encargada de informar a la ciudadanía ecuatoriana sobre el clima, tiempo y recursos hídricos para la protección de la sociedad y sus bienes. El INAMHI es miembro de la Organización Meteorológica Mundial (OMM), la cual cuenta con un equipo de profesionales técnicos y especializados en la materia para contribuir al desarrollo económico-social del Ecuador.

Actualmente, esta institución cuenta con la tecnología necesaria para dar seguimiento y predecir el comportamiento y estado de la atmósfera y las aguas interiores. A través de la difusión de la información meteorológica se pueden emitir diversas alertas tempranas, con el fin de reducir daños en los bienes materiales, proteger el ambiente y salvaguardar las vidas humanas ante desastres naturales o cambios climáticos extremos.

Por otra parte, gracias a los sistemas de transmisión como streaming y las nuevas infraestructuras de redes también es posible difundir contenido meteorológico audiovisual, lo que permite a la ciudadanía y a los diversos actores tener un manejo más fácil de la información, respetando la integridad y seguridad de los datos. Así, a través de una aplicación desarrollada en JAVA y utilizando la tecnología Google Web Toolkit, se han creado interfaces de usuario conectadas con Oracle para gestionar el contenido multimedia como audios y videos. De igual manera, empleando un servidor de streaming denominado Helix Universal Server 11 es posible brindar información por medio de la web consultando una base de datos de Oracle (Jorge Roberto Jova Rodríguez, 2019)

(Wang, Zhang, y Ma, 2017) En su trabajo de investigación explican que Python es un lenguaje de alto nivel que consiste en una sintaxis muy sencilla de comprender, facilitando el desarrollo

de software gracias a sus librerías que implementan funciones muy útiles. Para desarrollar aplicaciones de streamer se puede usar librerías como Gstreamer, que es un marco de desarrollo para crear aplicaciones de transmisión de medios como reproductor multimedia. Se pueden construir fácilmente reproductores multimedia muy buenos sobre Gstreamer, especialmente cuando se usa el objeto de alto nivel llamado playbin, muchas de las virtudes del marco GStreamer provienen de su modularidad. GStreamer puede incorporar sin problemas nuevos módulos de complemento.

(Han, Go, Noh, y Song, 2019) su estudio tuvo como objetivo proporcionar un servicio de transmisión de video en vivo de alta calidad, mediante un sistema de transmisión adaptable HTTP servidor-cliente cooperativo que codifica de forma adaptativa el segmento de vídeo para mejorar la utilización del ancho de banda. Los resultados de la simulación muestran que el sistema propuesto puede proporcionar una mayor utilización del ancho de banda y una menor diferencia de calidad que los sistemas de transmisión adaptativa HTTP existentes.

(Liu y Sun, 2012) en su investigación propone la utilización de la tecnología Web Socket como una solución para la comunicación web en tiempo real entre el cliente y el servidor. Demostrando que Web Socket puede disminuir el tráfico de red y la latencia en gran medida. Las pruebas de rendimiento realizadas en su investigación dan como resultado que la implementación del protocolo Web Socket es mucho más eficiente que el protocolo HTTP para la transmisión de datos en tiempo real.

Problema

Desde los inicios de la meteorología el ser humano ha desarrollado técnicas para estudiar y entender los fenómenos meteorológicos que se desarrollan en la atmósfera terrestre con la intensión de mitigar los riesgos naturales. Sin embargo, aunque existan los conocimientos básicos y la información receptada de radares, satélites y estaciones meteorológicas, es necesario contar con una tecnología capaz de transmitir esta información en tiempo real con el objetivo de reducir los efectos adversos sobre la población, los bienes, los servicios y el medio ambiente.

La Organización Meteorológica Mundial (OMM) promueve e incentiva a los Centros Meteorológicos e Hidrológicos de los países miembros, al mejoramiento continuo de las herramientas científicas y tecnológicas para que puedan ofrecer servicios de pronósticos meteorológicos, climáticos, hidrológicos y medioambientales cada vez más precisos y de mejor utilidad con el objetivo de dar mejores herramientas en la toma de decisiones para la protección de sus habitantes ante desastres naturales.(Tonato, 2021)

Según un informe de la Organización de Naciones Unidas (ONU) de la oficina para Asuntos Humanitarios "América Latina y el Caribe son las regiones más propensas a desastres naturales en el mundo, con 152 millones de personas afectadas por 1205 desastres entre los que se encuentran inundaciones, huracanes y tormentas, terremotos, sequías, aludes, incendios, temperaturas extremas y eventos volcánicos desde el año 2000".(*América Latina y el Caribe: la segunda región más propensa a los desastres | Noticias ONU*, 2020)

En el Ecuador los eventos hidrometeorológicos han ocasionado una variedad de impactos, siendo las inundaciones las más frecuentes y las que han provocado mayor afectación debido a su topografía irregular y ubicación geográfica. El Ecuador se encuentra ubicado dentro del cinturón de bajas presiones atmosféricas donde se sitúa la zona de convergencia intertropical (ZCIT), esta zona está formado por la convergencia de aire cálido y húmedo que al contacto

con masas de aire frio provenientes de las mesetas andinas o de las cimas altas de las montañas provocan precipitaciones de diversa magnitud. (*GEO Ecuador 2008 : informe sobre el estado del medio ambiente*, 2008)

Según un informe del Servicio Nacional de Gestión de Riesgos y Emergencias (SNGR), en el año 2021 el Ecuador registró 671 eventos peligrosos producto de las intensas lluvias, dejando como resultado 23964 personas afectadas, 5372 viviendas afectadas y 30205 metros de vías de acceso afectadas.(*Informe N°11 de situación época lluviosa Ecuador*, 2021)

Actualmente el Instituto Nacional de Meteorología e Hidrología (INAMHI) responsable en el Ecuador de la generación y difusión de información hidrometeorológica, cuenta con un sistema de radares de área local LAWR en banda X como herramienta complementaria a las ya existentes para implementar pronóstico y monitoreo hidrometeorológico. Los radares se sitúan en tres puntos estratégicos de la ciudad de Quito: Mitaloma, El Troje y Monjas, al momento se encuentra un solo radar operativo (Monjas) el cual genera una imagen de la precipitación en un radio de 60 km.(Tonato, 2021)

El presente proyecto técnico responde a la siguiente problemática ¿Cómo se puede mejorar la transmisión de imágenes meteorológicas emitidos por el radar meteorológico Furuno?. El Instituto Nacional de Meteorología e Hidrología (INAMHI) cuenta con un software que permite la visualización de imágenes meteorológicas generada por el radar de banda X Furuno. La visualización de imágenes meteorológicas es de suma importancia en la toma de decisiones oportunas para la protección de sus habitantes ante fenómenos meteorológicos. De tal manera al mejorar los tiempos de transmisión de las imágenes meteorológicas aumentara la posibilidad de pronosticar el clima.

Entre las diferentes causas que pueden estar originando este problema, hemos detectado que un software obsoleto afecta el rendimiento y la velocidad de una aplicación, además las nuevas

tecnologías para la transmisión de contenido multimedia (streaming) tienen mayor compatibilidad con software actualizado, garantizan un correcto funcionamiento y mejoran la experiencia del usuario al momento de utilizarlo.

Por ello, el presente proyecto técnico pretende, generar una aplicación de streaming en vivo para la visualización de imágenes obtenidas del radar meteorológico, utilizando tecnologías modernas, estables y escalables con un modelo de desarrollo open source.

Justificación

La información obtenida sobre el funcionamiento del radar, el software de lectura y procesamiento de imágenes, y la página web para visualización de la información meteorológica, se encuentra actualmente con fallas y fuera de funcionamiento, es por tal motivo que el Instituto Nacional de Meteorología e Hidrología (INAMHI) busca un solución tecnológica para crear un nuevo sistemas web que integre nuevas tecnologías Open Source, compatibles con la mayoría de sistemas operativos y navegadores web, para un monitoreo más preciso de la situación del clima en Quito en tiempo real.

La importancia del presente proyecto radica en contar con un sistema de monitoreo climático en tiempo real del distrito metropolitano de Quito, que brinde al departamento de meteorología del INAMHI la facilidad de dar informes y/o predicciones más acertadas sobre los cambios climáticos de la capital, alertando de manera más oportuna de posibles desastres naturales.

Lo que se busca implementar con el presente proyecto es el envío de imágenes emitidas por el radar en tiempo real, por medio de streaming, tal propuesta pretende disminuir el tiempo de visualización de las imágenes en el sitio web, la propuesta es realizar el proyecto con herramientas y software vanguardistas para que pueda ser escalable y mantenible en el tiempo. Como estudiantes de la rama de desarrollo, este proyecto permite poner en práctica cada una de las

destrezas adquiridas en la entidad educativa.

Objetivo General

Desarrollar una aplicación de streaming que permita mejorar la transmisión de datos recibidos por el radar Furuno, en la ciudad de Quito.

Objetivo Específicos

- Investigar el funcionamiento y arquitectura del radar Furuno.
- Procesar la información recibida del radar para mostrarlas en web.
- Implementar un back-end que se conecte al servidor del radar usando Python y un front-end en Angular para visualizar la información.

Metodología

La metodología ágil que se utilizará para el desarrollo de este proyecto será Kanban debido a que es considerado como un sistema de producción eficiente y efectivo ya que gestiona la realización de tareas desde su planteamiento hasta la finalización de esta. Kanban es una herramienta de desarrollo que permite el aprendizaje y la gestión al optimizar el flujo de trabajo dentro de un proceso. (Klipp, 2014)

Según (Patilla y cols., 2021) es una herramienta que ofrece mejorar la eficiencia del flujo de trabajo. De este modo, todos los equipos pueden visualizar las tareas en proceso y esto proporciona una transparencia general en la organización, ya que permite a los miembros de los diferentes equipos tener una clara visión del flujo de desarrollo y facilita mantener el control de las diferentes etapas. Las cartas Kanban ayudan identificar las tareas que estaban por hacer, las

que se están haciendo y las que ya se hicieron. Kanban se basa en un desarrollo incremental, es decir que el trabajo se divide en varias partes. Por lo tanto, no se trata una tarea en sí, se trata de agilizar el proceso dividiendo la tarea en varios pasos. Esto permite controlar el flujo y seguimiento del proyecto.

KANBAN: Su nombre viene del japonés “kan” (visual) y “ban” (tarjetas), esta metodología cumple como objetivo gestionar de manera general como se van completando las tareas. Las ventajas principales de esta metodología son: fácil de utilizar, actualizar y asumir por parte del equipo, lo más destacable es que es muy visual (Gilibets, 2020). Para aplicar la metodología Kanban hay que generar un tablero de tareas que permita mejorar el flujo de trabajo.

Capítulo 1

Fundamentos teóricos

1.1 Meteorología

La Meteorología es la ciencia encargada del estudio de la atmósfera terrestre (tropósfera), responsable del clima atmosférico, además de sus propiedades y los fenómenos que se suscitan dentro de ella. Su estudio se basa en el conocimiento de una serie variables meteorológicas (magnitudes), como la presión, el viento, la humedad, la precipitación, entre otras, las cuales varían tanto en el espacio como en el tiempo. (*Meteorología y climatología : unidad didáctica : Semana de la Ciencia y la Tecnología 2004*, 2004)

1.1.1 Variables meteorológicas

Las principales variables climáticas que describen las condiciones meteorológicas se enumeran a continuación.

1.1.1.1 Presión atmosférica

La presión atmosférica normal es el peso de aire seco y puro sobre una unidad de superficie (1 cm^2) medida al nivel del mar, a una latitud de 45° y a una temperatura de 0°C. En meteorología, una presión más alta de lo normal se llama presión alta, una presión más baja se llama presión baja.(*Meteorología y climatología : unidad didáctica : Semana de la Ciencia y la Tecnología 2004*, 2004)

La presión atmosférica normal se ha utilizado durante mucho tiempo como unidad de medida (1 atm) y equivale al peso de una columna de mercurio de 760 mm de altura, con una sección de 1 cm^2. Sin embargo, la presión en meteorología actualmente se mide en hectopascales (hPa).

La presión atmosférica normal es igual a 1013 hPa:

$$1atm = 1013hPa = 1013mbar = 760mmHg$$

1.1.1.2 Viento

El viento es el fenómeno atmosférico de desplazamiento, principalmente horizontal, de masas de aire, normalmente se origina cuando entre dos puntos se establece una cierta diferencia de presión o de temperatura. (Fortelli, 2021)

Cuando existe una diferencia de presión entre dos zonas geográficas, el aire ubicado en la zona de alta presión se desplazará hacia la zona de baja presión, por otro lado, cuando la temperatura de una masa de aire es superior a su entorno, su volumen tiende aumentar y su densidad a disminuir, provocando que la masa de aire caliente ascienda dejando un vacío que posteriormente será ocupado por una masa de aire frío, en su desplazamiento ocasionarán el viento.(*Meteorología y climatología : unidad didáctica : Semana de la Ciencia y la Tecnología 2004*, 2004)

1.1.1.3 Humedad

La humedad atmosférica es la cantidad de vapor de agua contenido en la atmósfera, la cantidad de vapor depende de diversos factores tales como la evaporación de aguas superficiales, del agua de los suelos y de la transpiración de las plantas.(Andrades Rodríguez y Múñez León, 2012)

La medida de humedad que más se utiliza es la humedad relativa, se entiende por humedad relativa al cociente entre la cantidad de vapor de agua contenido en la atmósfera y la máxima que podría contener, este valor resultante es la capacidad que tiene una masa de aire para almacenar vapor agua y se lo expresa en tanto por ciento (%) .(*Meteorología y climatología : unidad didáctica : Semana de la Ciencia y la Tecnología 2004*, 2004)

Cuando el porcentaje de humedad relativa supera el 100 %, el excedente de vapor de agua almacenado en la masa de aire desciende a través de precipitaciones líquidas o sólidas, según las condiciones ambientales. (Andrades Rodríguez y Múñez León, 2012)

1.1.1.4 Nubosidad y precipitación

Una nube esta formado por un conjunto de diminutas gotas de agua o de pequeños cristales de hielo que se encuentran suspendidos en la atmósfera, procedentes de la condensación del vapor de agua sobre partículas microscópicas.(Andrades Rodríguez y Múñez León, 2012)

La condensación del vapor de agua puede estar provocada por cualquier proceso atmosférico que produzca un enfriamiento en una masa de aire:

1. Por el cambio de temperatura que puede tener una masa de aire.

2. Cuando existe el desplazamiento de una masa de aire.

Cuando las gotas de agua o cristales de hielo alcanzan un peso superior al soportado por la fuerza de arrastre, las gotas o cristales de hielo caerán hacia el suelo produciendo la lluvia, el granizo o una mezcla de ambas, dando lugar a la precipitación.

1.2 Radar

Para el funcionamiento de un radar es necesario revisar lo que el físico ingles James Maxwell desarrolló, las ecuaciones que permiten entender el comportamiento de las ondas electromagnéticas. Posteriormente el físico alemán Heinrich Herz demostró, partiendo de las ecuaciones de Maxwell, las leyes de reflexión de las “ondas de radio”. Logrando así demostrar que hay ciertas propiedades físicas en el medio ambiente que facilita la reflexión de estas ondas. Aleksandr Stepánovich Popov, un científico ruso, luego de realizar varios experimentos hacia finales del

siglo XIX con emisores y receptores documentó cada uno de estos y llegó a la conclusión de la posibilidad de detectar objetos por medio de ondas de radio (Quiroga, 2018)

Para el año de 1895 un inventor joven italiano de alrededor 21 años llamado Guglielmo Marconi recopiló la teoría electromagnética de Maxwell y los conocimientos de los experimentos de Herz, el joven logró establecer una transmisión de telegrafía sin hilos que de hecho fue la primera de la historia. (Sanfuentes, 2000). En 1920 Marconi retomando los estudios hizo notar que era posible que las ondas de radio (de hasta 30 MHz) dirigidas hacia un haz, estas se reflejan por un objeto, como un buque, de este modo se obtiene su presencia, distancia, sin importar las condiciones climáticas, noches oscuras o días llenos de neblina. (Sanfuentes, 2000)

El radar es una tecnología que utiliza ondas de radio para detectar objetos, esto permite obtener información importante sobre dichos objetos. Su desarrollo se remonta a comienzos del siglo XX y principalmente a partir de la tercera década de dicho siglo. (Quiroga, 2018). Durante la década de 1930 varios países como Alemania, Estados Unidos, Francia, Holanda, Italia, Japón, el Reino Unido y la URSS invirtieron tiempo y dinero en la investigación y desarrollo de detección de objetos por medio de ondas de radio. (Quiroga, 2018). Para el desarrollo de los componentes de los radares en un principio se basó en tecnologías existentes orientadas a usos diferentes a la detección de objetos a la distancia. Múltiples estudiosos contemporáneos comenzaron a encontrar maneras de implementar estos artefactos para el uso militar, como en aviones y mantener alerta a los pilotos de la presencia de tormentas, con el avance de la tecnología de la década, se utilizaba con tubos de rayos catódicos a partir de 1923 (Sanfuentes, 2000). En el quinto año de la tercera década del siglo XX Watson Watt junto al instituto de estudios de radio logró la primera detección de un avión a través de ondas de radio a una distancia de 15 millas, posteriormente el mismo año utilizando ondas de 12 MHz se detectó un bombardero que se encontraba a una distancia de 40 millas, este nuevo dispositivo fue bautizado como “Radio Detection Finding”

(RDF) pero solía tener el sobrenombre de “rayo de la muerte”. Tiempo más tarde el dispositivo se lo renombró a “Radio Detection and Ranging” (Radar) (Sanfuentes, 2000)

El radar es una invención que, si bien estuvo enfocada en un principio para fines bélicos, actualmente se tiene una amplia gama de diferentes tipos que se utilizan para funciones variadas, en principio todos tienen el mismo funcionamiento, emitir una onda de radio a X frecuencia en MHz para de este modo esperar el rebote de esta y obtener información de la onda. Entre los radares de “uso diario” podemos destacar los que detectan aviones ligeros, los utilizados en las autopistas para medir la velocidad de los vehículos y en el caso de este trabajo de investigación, se hace uso de un radar meteorológico.

1.2.1 Radar meteorológico

Uno de los tipos de radar más aplicados es el “Local Area Weather Radar” (Radar meteorológico de área local) o mejor conocido como LAWR en sus siglas en inglés. Este tipo de radar se produce a partir de un radar marino de banda X, esto hace que su costo se más accesible en comparación a un radar meteorológico convencional. Los radares meteorológicos de banda X, C o S se basan en una relación teórica entre la emisión de energía del radar, la reflectividad de onda y la intensidad de lluvia pero un modelo LAWR tiene una relación empírica entre la reflectividad de onda y la intensidad de lluvia, esto se debe a sus limitaciones de diseño dado que es un modelo marino.

El radar meteorológico utilizado como objeto de estudio para este proyecto emite una señal en un conjunto de frecuencias repartidas en un espectro electromagnético de 10 MHZ hasta los 600 GHz. La operación del radar consiste en la emisión de ondas electromagnéticas, dichas señales (dependiendo de su frecuencia) al chocar con partículas de diferentes tamaños pueden ser: absorbidas, reflejadas, refractadas o difractadas.

Las ondas reflejadas (ecos) son recibidas de nuevo por el radar y brindan información de distancia, tamaño y velocidad. Los radares meteorológicos funcionan utilizando tecnología de pulso la cual generalmente existe un generador de microonda (magnetrón), una guía de ondas y una antena parabólica. Las longitudes de onda emitida van desde 1 cm a 10 cm, es decir aproximadamente diez veces el diámetro de las gotas de lluvia u otras partículas de interés. El concepto del radar consiste en que parte de la energía de cada pulso rebota en las partículas de la atmósfera, estas ondas rebotan y regresan al lugar de donde fue emitida. Actualmente existen diversos tipos de radares con diferentes configuraciones diseñado para el tipo de partícula a ser analizada (granizo, lluvia, niebla, etc.). Cabe destacar que existe un rango de frecuencias específicas para los radares meteorológicos, este rango es asignado por la Unión Internacional de Telecomunicaciones (ITU), estas son: 2.8Ghz (Radar de banda S), 5.6Ghz (Radar de banda C), 9.4Ghz (Radar de banda X) y 35.6Ghz (Radar de banda Ka).

Las longitudes de onda más costar se utilizan para analizar partículas diminutas, el problema que surge con estas ondas es que la señal se va atenuando más rápido. Así que un radar de banda X se utiliza para analizar partículas en distancias muy cortas mientras que un radar de banda Ka se usa solo para estudiar fenómenos producidos en pequeñas partículas como la niebla. En condiciones normales el radar de banda X funciona en un radio de 60km para pronóstico y alerta de lluvias y un radio de 20 a 30 Km para medición de intensidad de precipitación.

1.2.2 Transmisión de imágenes

Para realizar el monitoreo meteorológico, el INAMHI cuenta con la siguiente infraestructura como se puede observar en la figura 1, esta infraestructura es utilizada para el envío de imágenes que posteriormente se utilizan sobre un mapa de la ciudad de Quito, todo esto se realiza con la finalidad de monitorear el clima de la ciudad, la cantidad de precipitaciones, dirección de la

lluvia y cantidad de agua que podría caer, de esta manera se puede determinar qué sectores pueden llegar a ser afectados por estos fenómenos meteorológicos.

Infraestructura de comunicación del radar con servidor INAMHI

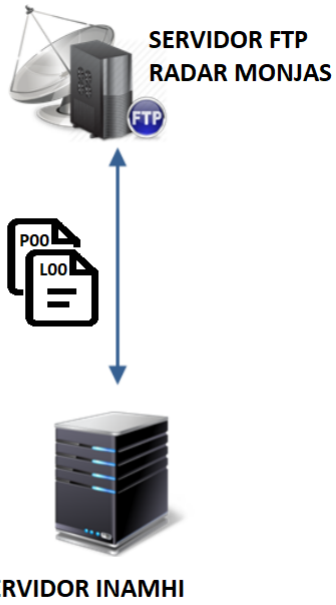


Figura 1: Infraestructura de comunicación del radar Furuno con el servidor INAMHI para el envió de información meteorológica.

Elaborado por: Kevin Changoluisa, Bryan Imbaquingo

- El radar está conectado punto a punto a un servidor en el INAMHI.
- Permite conexión por escritorio remoto y FTP.
- Desde el servidor del INAMHI se reciben los archivos archivos .p00 que serán procesados para mostrarse al cliente desde el backend.

1.3 Procesamiento de datos meteorológicos

Para generar un pronóstico meteorológico, se necesita de la información procesada del radar también llamada imagen meteorológica, lo cual permite la estimación de la precipitación acu-

mulada en el área que opera el radar de manera gráfica.

El procesamiento de los datos meteorológicos del radar está compuesto por dos partes:

- Procesamiento de señal
- Procesamiento de información meteorológica

1.3.1 Procesamiento de señal

El procesamiento de señales es controlada por el computador para procesamiento de señales, en donde la señal de voltaje digitalizada (señal de video) es procesada teniendo en cuenta los valores de reflectividad también conocidos como cuentas. Durante el procesamiento de la señal el dato atraviesa varios esquemas con el propósito de manejar los obstáculos bien conocidos por el radar. (Llugsi Cañar, 2013)

Las diferentes etapas de procesamiento de señal se muestran a continuación

Diagrama de flujo del procesamiento de señales

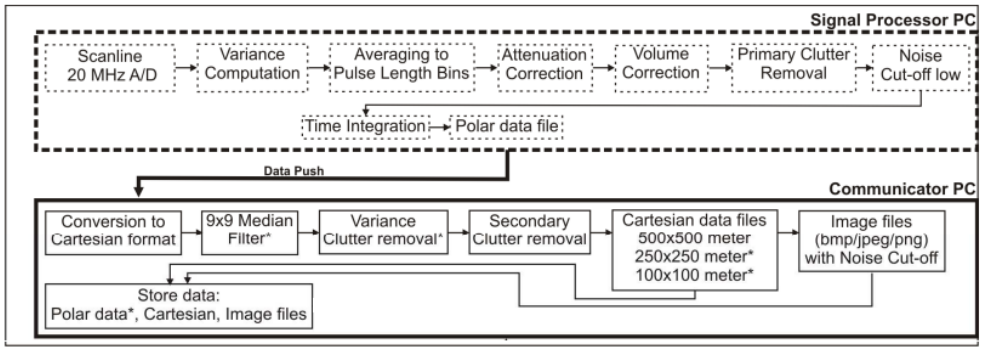


Figura 2: Procesamiento de señal radar Furuno.
Fuente: (Niels, 2010)

1.3.2 Procesamiento de información meteorológica

La información resultante del procesamiento de señal del radar Furuno son dos archivos binarios, para los datos polares la extensión .L00 y para los datos cartesianos la extensión .P00, ambos archivos contienen metadatos acerca de la información del radar y la medición cuantitativa de la reflectividad a lo largo del eje escaneado.(Niels, 2010)

Como se muestra en la figura 3 la información del archivo binario P00, se encuentra representada mediante una matriz de dimensiones NxM, en donde cada elemento representa un píxel y a su vez el pixel representa las condiciones de una región geográfica, cada elemento toma un valor discreto entre 0 y 254. El radar está ubicado en el centro de la matriz.

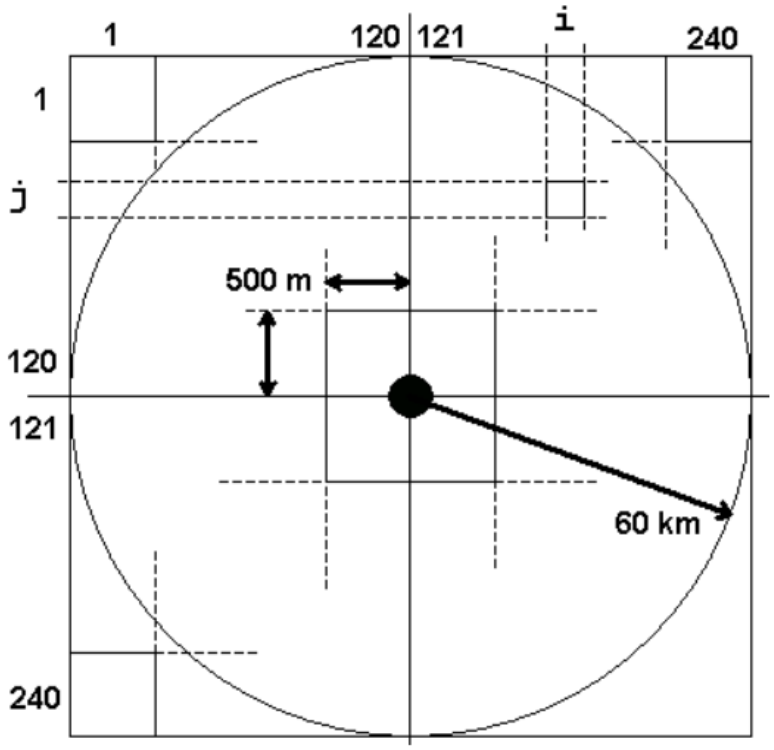


Figura 3: Representación de los datos cartesianos de los archivos P00.
Fuente: (Niels, 2010)

1.3.3 Imagen

Una imagen es una representación bidimensional de una escena, un objeto, un ser vivo o incluso un concepto, que se encuentra en el mundo tridimensional. La imagen es una codificación numérica que contiene la información del espectro electromagnético de una señal captada por un sensor. (Alegre Gutiérrez, Pajares Martinsanz, y de la Escalera Hueso, 2016)

1.3.4 Imagen digital

La imagen digital se representa mediante una matriz bidimensional de dimensiones N x M, donde cada elemento representa un punto o píxel. Cada píxel contiene un valor discreto que cuantifica la intensidad de la luz, desde niveles oscuros de luminosidad hasta valores claros. (Alegre Gutiérrez y cols., 2016)

En la figura 4 vemos la representación gráfica en escala de grises de una matriz bidimensional compuesta por sus elementos, en donde cada elemento representa un píxel con un valor discreto en un rango de 0 a 255.

Matriz bidimensional en escala de grises

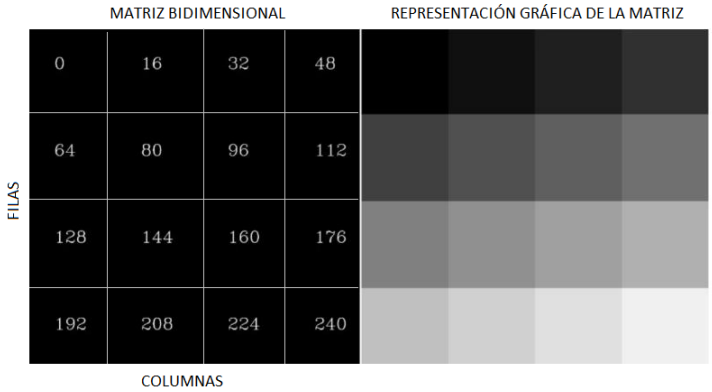


Figura 4: Representación gráfica de una matriz bidimensional en escala de grises.
Elaborado por: Kevin Changoluisa, Bryan Imbaquingo

1.3.5 Procesamiento de imágenes digitales

El procesamiento de imágenes digitales es un conjunto de técnicas que implican la alteración del valor de cada píxel con el objetivo de mejorar la calidad de la imagen para mejorar la representación de su información. Su origen se da gracias a los avances tecnológicos para captar y manipular grandes cantidades de información espacial representada en matrices.(da Silva y Mendonça, 2005)

Existe tres pasos básicos para el procesamiento de imágenes que son los siguientes:

- Importar la imagen: Es la lectura de la imagen a través de un software que interprete sus valores y permita modificarlos.
- Análisis y manipulación la imagen: Son técnicas tales como la reducción de ruido y realce de detalles aplicando operaciones matemáticas a la representación matricial de la imagen.
- Resultado: Es una nueva imagen o el informe que se basa en el análisis de la imagen original con la manipulada.

1.3.6 Ruido de la imagen

El ruido es la información contaminada de la imagen, su contaminación se debe a factores externos o internos que intervinieron al momento de la adquisición. En una imagen meteorológica el ruido hace referencia a la dispersión de la señal en el momento de escaneo debido a la interferencia de ondas electromagnéticas. La magnitud del ruido de la imagen puede variar desde manchas casi imperceptibles a manchas que dificulten la interpretación de la imagen.

Existen modelos de ruido de acuerdo a su densidad e intensidad, ruido Gaussiano(normal), uniforme, impulsivo (sal y pimienta) entre otros. Para poder suavizar el ruido se presenta algoritmos con técnica de filtrado que implementan métodos para obtener información de las imágenes que son mostradas en diferentes escalas, además se puede ocultar valores anómalos. (Nachtegael, Van der Weken, Van De Ville, y Kerre, 2003)

- Ruido Gaussiano: es causado por la exposición del sensor de la cámara a intensidades de luz muy bajas y/o altas, durante la transmisión, o como el ruido provocado por el circuito electrónico. Las técnicas que se emplean durante el filtrado espacial convencional diseñados para la eliminación de ruido incluyen: el filtrado mediano (de la mediana) y el desenfoque Gaussiano.(da Silva y Mendonça, 2005).

- Ruido de sal y pimienta: También se conoce como ruido impulsivo. Este ruido puede ser causado por perturbaciones agudas y repentinas en la señal de la imagen. Aparece como píxeles raros en blanco y negro. Un método efectivo para reducir este tipo de ruido es un filtro mediano o un filtro morfológico (da Silva y Mendonça, 2005).

- Ruido de disparo: El ruido de disparo se muestra en las partes más brillantes de una imagen, suele ser el causado por fluctuaciones cuánticas estadísticas, es decir, la variación en el número de fotones detectados en un nivel de exposición dado. Es un tipo de ruido que se representa como un proceso de Poisson (da Silva y Mendonça, 2005).

- Ruido uniforme: El ruido causado por la cuantificación de píxeles de una imagen detectada en varios niveles discretos se conoce como ruido uniforme. Tiene una distribución aproximadamente uniforme (da Silva y Mendonça, 2005).

En las imágenes meteorológicas generadas por radar el patrón más común observado es el de ”granulado”, que es un efecto que se observa en forma de una cantidad de puntos blancos y negros (efecto sal y pimienta) sobre la imagen, esto debido a la interferencia en la onda del radar, como tal se muestra la figura 5 antes de su tratamiento y procesamiento. (Guerra, 2002)

Ruido en imágenes meteorológicas

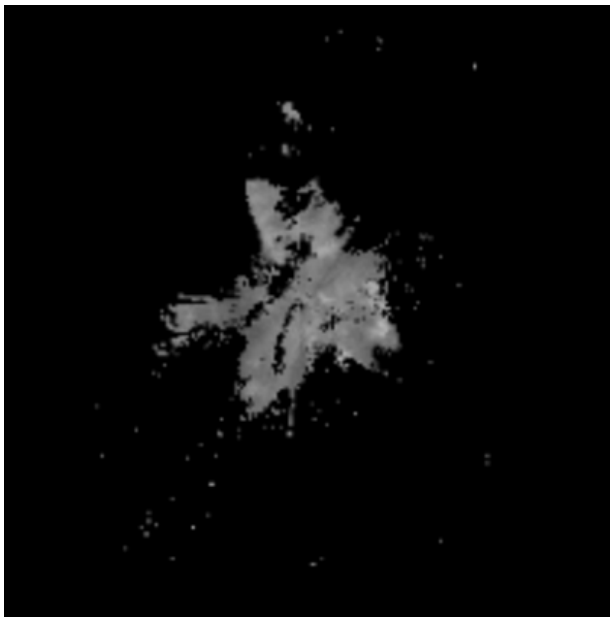


Figura 5: Imagen meteorológica radar Furuno Quito-Ecuador 01/01/2018.
Elaborado por: Kevin Changoluisa, Bryan Imbaquingo

1.3.7 Filtrado de la imagen

El filtrado es una operación de vecindad que consiste en generar un nuevo valor a los pixeles de la imagen mediante una función del valor original y de los pixeles circundantes. La vecindad se define como la relación que tiene un píxel con el conjunto de píxeles que lo rodean.

En la figura 6 se presenta los dos tipos de vecindad que posee un pixel, vecindad de cuatro u ocho vecinos.

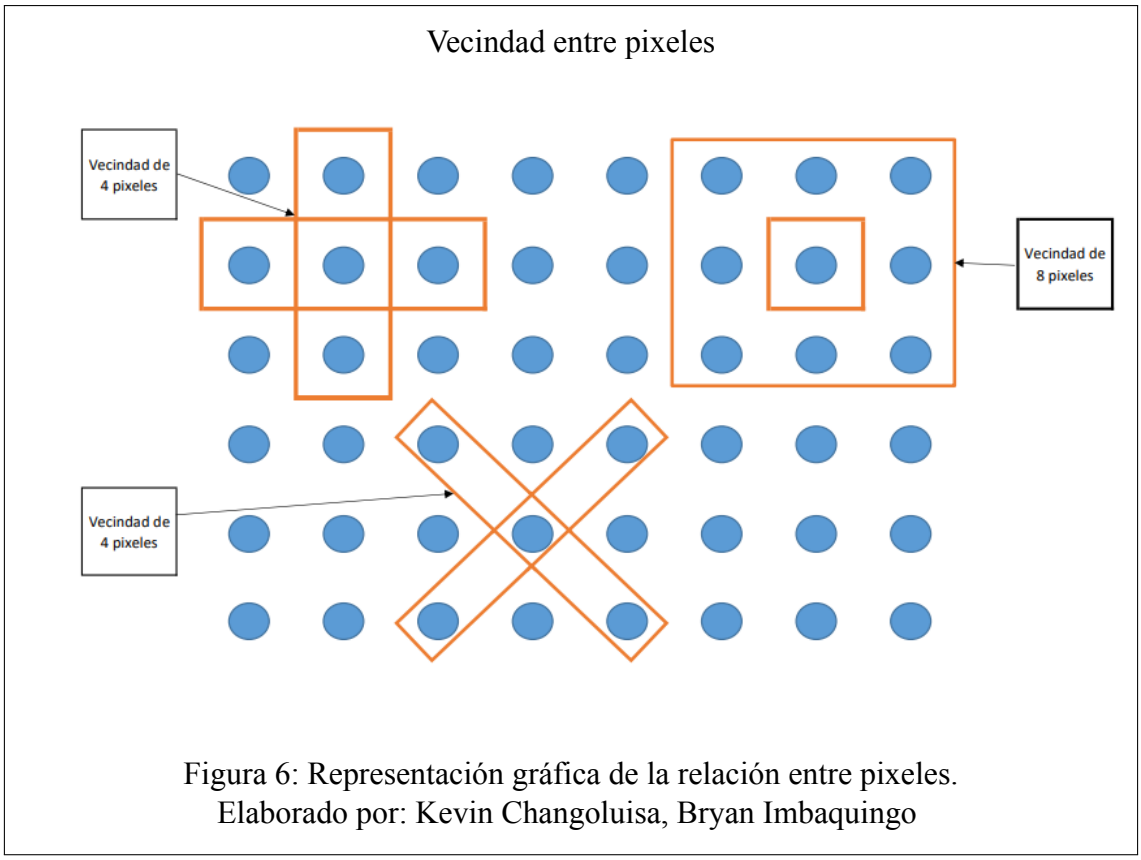


Figura 6: Representación gráfica de la relación entre pixeles.
Elaborado por: Kevin Changoluisa, Bryan Imbaquingo

Los filtros más utilizados para suavizar imágenes son los de paso bajo, son útiles para eliminar el ruido que aparece en la imagen. También se utiliza para resaltar la información importante de la imagen.(Sarría, 2006)

Existen varias posibilidades:

- Filtro de la media, es una técnica de procesamiento de señal útil para suprimir el ruido de la imagen, el pixel central se reemplaza por la media de todos los pixeles de la vecindad. El tamaño de la vecindad debe ser impar para que exista este elemento central.

- Filtro de media ponderada, es utilizado para obtener un resultados más parecido a la imagen original, evitando un resultado borroso en la imagen, esta técnica de filtrado asigna diferentes pesos a los píxeles de la vecindad y busca el promedio ponderado de los píxeles.

- Filtro mediano es una técnica de procesamiento de señal no lineal, reduce el efecto borroso que tienen las imágenes, ideal para eliminar el ruido tipo sal y la pimienta. El filtro

mediano se usa para eliminar el desorden residual, es decir, todos los valores de píxeles se reemplazan por la mediana de sus vecinos.

Un efecto del filtro mediano es que casi todos los píxeles individuales o adyacentes en diagonal que tienen un eco se eliminan y se restablecen con un valor de cero. En las imágenes meteorológicas esto es aceptable ya que la gran mayoría de estos píxeles tienen desorden o precipitación insignificante.(Niels, 2010)

En la figura 7 se puede observar la diferencia entre una imagen meteorológica sin filtro mediano y con filtro mediano.

Imagen meteorológica de precipitaciones

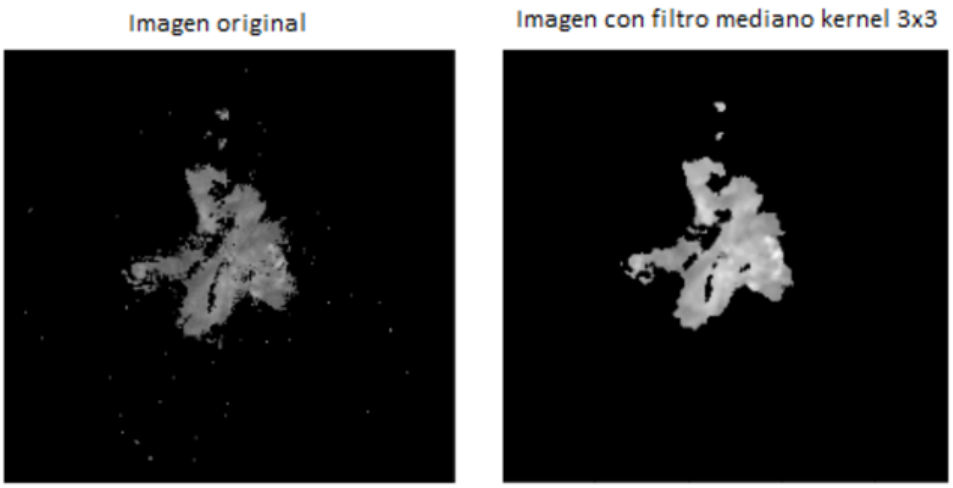


Figura 7: Imagen meteorológica radar Furuno Quito-Ecuador 01/01/2018.
Elaborado por: Kevin Changoluisa, Bryan Imbaquingo

1.4 Herramientas de desarrollo

1.4.1 Software Libre

El término software libre o ”free software” en inglés, identifica una categoría muy específica de aplicaciones de software, caracterizada porque la licencia de uso con la que se distribuyen

garantiza siempre al menos cuatro libertades fundamentales: libertad de ejecutar el programa, libertad de estudiar y modificar el programa, libertad de redistribuir copias del programa, libertad para mejorar y distribuir el programa.(Stallman, 2004)

En Ecuador, el modelo de Software Libre se convierte en una política de estado, garantizando los cuatro pilares del software libre y priorizando de manera absoluta la adquisición software libre mediante decreto ejecutivo No. 1014 emitido el 10 de abril de 2008 en el cual se dispone "El uso de Software Libre en los sistemas y equipamientos informáticos de la Administración Pública de Ecuador. Es interés del Gobierno ecuatoriano alcanzar soberanía y autonomía tecnológica, así como un ahorro de recursos públicos."(*Estrategia para la implantación de software libre en la administración pública central*, 2009)

Algunas ventajas del software libre son:

- El software puede adaptarse más rápido a las necesidades del usuario.
- La disponibilidad del código fuente favorece la corrección de errores o fallas de seguridad y previene conductas maliciosas ocultas.
- La posibilidad de modificación favorece la evolución del software.
- Reduce costos de licencias

El software libre se distribuye con licencias de usuario que definen los derechos y obligaciones de los usuarios. La licencia más conocida es la GNU GPL (General Public Licence), pretende garantizar su libertad para compartir y cambiar todas las versiones de un programa, para asegurarse de que siga siendo software libre para todos sus usuarios. La última versión de esta licencia es la 3.(*licenses-gnu*, s.f.)

La licencia establecida para la aplicación web, así como para el software de procesamiento de información meteorológica y el software para la transmisión de información WebSocket,

será GNU GPL versión 3, de tal manera se garantiza que el software será libre y seguirá siendo software libre, sin importar quién cambie o distribuya el programa.

1.4.2 Lenguaje de programación

Un lenguaje de programación es una serie de instrucciones traducidas de un lenguaje normal a un lenguaje máquina, respetando un conjunto de reglas de sintaxis y de semántica para llevar acabo un correcto calculo en el ordenador. (Mihaela Juganaru, 2014)

El lenguaje de programación permite una comunicación con las computadoras a través de una sintaxis comprensible para el ser humano llamado código fuente, que la computadora entiende e interpreta en lenguaje de máquina mediante un compilador.

En la historia de la computación han sido propuestos varios lenguajes, en la figura 8 se visualiza el orden cronológico de los lenguajes de programación a través de la historia.

Historia de los lenguajes de programación

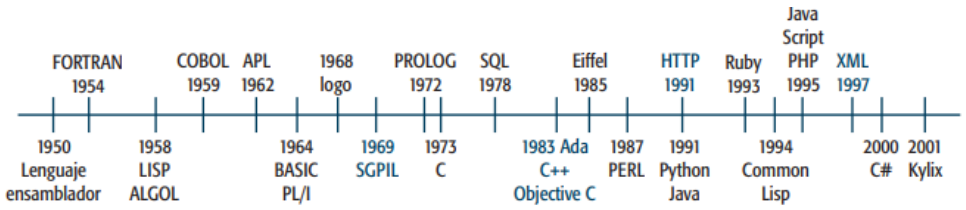


Figura 8: Línea de tiempo de los lenguajes de programación.
Elaborado por: (Mihaela Juganaru, 2014)

En la actualidad se trabaja en la creación o mejora de lenguajes de programación capaces de resolver las problemáticas actuales, definiendo nuevos estándares.

Actualmente existen varios lenguajes de programación multiparadigma para diferentes propósitos que pueden ser utilizados en el backend (lado del servidor) y/o en el frontend (lado del cliente), estos a su vez con la ayuda de librerías o frameworks dedicados ayudan al desarrollador a no tener que realizar todo desde absoluto cero, instalando librerías o creando un proyecto

nuevo con un framework, ya viene integrado con las herramientas necesarias para un área en específico.

1.4.3 Backend

El backend es la parte que no se puede ver a simple vista y lleva la lógica del negocio o servicio, este se encuentra del lado del servidor procesando la información o enviado las peticiones que se realizan desde el lado del cliente, esta es la parte robusta y lógica de cualquier sistema o aplicación web. Para el desarrollo de esta parte del proyecto se utilizó Python, ya que este lenguaje fue propuesto como requerimiento del cliente (INAMHI) para la realización del backend y la librería en la que se desarrolla la lógica del negocio es en Flask.

- Python: Python es un lenguaje de programación interpretado, multiplataforma y multiparadigma creado por Guido van Rossum en el año de 1989, su objetivo principal es la facilidad de aprendizaje para personas ajenas a las Ciencias de la Computación. (Monsálvez, 2017) Las principales características del lenguaje Pyhon son:
  - Sintaxis simple
  - Alta legibilidad
  - Multiparadigma
  - Software libre

  Python cuenta con dos tecnologías Backend muy populares:
  - Flask
  - Django

1.4.4 Frontend

De igual modo que el backend, el frontend tiene varias librerías o frameworks que ayudan al desarrollador con un conjunto de herramientas para realizar su trabajo de una manera más eficiente, entre las tecnologías más utilizadas actualmente tenemos: React, Angular y Vue.

El desarrollo del frontend se realizó utilizando el framework de Angular, ya que este framework fue propuesto por el cliente como requerimiento para el desarrollo de la aplicación.

1.4.4.1 Angular

Es un framework desarrollado en Typescript de código abierto, mantenido y soportado por Google para la creación de aplicaciones web de una sola página (single page app SPA), su principal objetivo es crear webs robustas que puedan soportar el patrón MVC (modelo vista controlador) y faciliten los test en el software. (Wohlgethan, 2018)

1.4.4.2 TypeScript

TypeScript es un lenguaje de programación desarrollado por Microsoft orientado en la filosofía Opensource. Básicamente es un superconjunto de JavaScript, que agrega tipos, clases, interfaces y módulos opcionales al JavaScript tradicional.(Fenton, 2017)

TypeScript es un lenguaje tipado, es decir, agrega definiciones de tipos estáticos que le permiten describir la forma de un objeto, documentarlo mejor y permitir que TypeScript verifique que el código funciona correctamente.(Fenton, 2017)

Uno de los puntos fuertes de TypeScript, que lo distingue de sus competidores CoffeeScript y Dart, es el hecho de que es un superconjunto: cualquier código escrito en JavaScript también es compatible con la sintaxis y la semántica de TypeScript.

1.5 Streaming

El streaming es una tecnología utilizada en las telecomunicaciones para la transmisión de contenido multimedia en tiempo real a través de Internet. De esta forma, la información perteneciente al contenido solicitado se reproducirá automáticamente a medida que llegue al dispositivo de destino.(Pérez, 2021)

1.5.1 Live Streaming

El live Streaming o transmisión en vivo, es la transmisión en tiempo real de contenido mediante una conexión de flujo continuo, se puede decir que es muy similar a la transmisión tradicional. En este caso, los datos se transmiten utilizando una o más técnicas de compresión apropiadas, de forma que se aligere al máximo la carga de la red utilizada para la transmisión de la información necesaria. Sin embargo, el uso de estas técnicas particulares de compresión implica un ligero retraso, del orden de unos diez segundos, en la transmisión del flujo de información requerido, sin embargo, en la mayoría de los casos, no constituye un problema para el receptor de dicha información. Ejemplos típicos son las transmisiones de eventos deportivos, conciertos o sistemas de monitoreo del clima.

1.5.2 Streaming on demand

El streaming on demand o transmisión bajo demanda, todos los contenidos multimedia que se pueden solicitar están listos para su uso, en forma de archivos comprimidos, en un ordenador particular diseñado para atender las solicitudes. En este caso, cuando un usuario solicita el contenido multimedia, es inmediatamente descomprimido por un programa o dispositivo apropiado, llamado códec, que reproduce inmediatamente la información contenida en el fichero solicitado unos segundos después del inicio de la recepción del mismo.

El retraso que se produce entre el inicio de la recepción del archivo y el inicio de su reproducción es denominado buffer, se utiliza para subsanar eventuales latencias o breves interrupciones debidas a la red utilizada para la transmisión del contenido buscado. Ejemplos típicos de transmisión a pedido son el popular sitio de YouTube o los servicios pagos más recientes llamados Netflix y Disney +.

1.6 WebSocket

WebSocket es un protocolo de red basado en TCP, que establece un canal full-duplex (ambos dispositivos pueden transmitir y recibir datos al mismo tiempo) y de baja latencia entre el servidor y el navegador, estableciendo conexiones bidireccionales para el intercambio de datos.(Melnikov y Fette, 2011)

Además, este protocolo cuenta con estados, lo que significa que la conexión entre el cliente y el servidor se mantendrá activa hasta que cualquiera de las partes (cliente o servidor) la finalice, después de cerrar la conexión por parte del cliente y el servidor, la conexión finaliza desde ambos extremos. Inicialmente WebSocket fue propuesto como parte de la especificación HTML5, que promete brindar facilidad de desarrollo y eficiencia de red a aplicaciones web modernas e interactivas. Ahora WebSocket se encuentra normalizado por Internet Engineering Task Force (IETF) como el estándar de comunicación RFC 6455.(Melnikov y Fette, 2011)

1.6.1 ¿Cómo funciona WebSocket?

Cada conexión WebSocket inicia como una solicitud HTTP, en la cual se especifica una operación de actualización en el encabezado que indica que el cliente desea actualizar la conexión a un protocolo diferente, en este caso a WebSocket. (Melnikov y Fette, 2011)

En la figura 9 se presenta un ejemplo de un protocolo de enlace del lado del cliente:

Solicitud HTTP del lado del cliente.

```
Request URL: "http://server-socketio-kevin.herokuapp.com/"
Request Method: "GET"
Status Code: "101 Switching Protocols"
Request Headers
 Sec-WebSocket-Version: "13"
 Sec-WebSocket-Key: "ZUfiNNYcylBhgEh59tg1CA=="
 Connection: "Upgrade"
 Upgrade: "websocket"
 Sec-WebSocket-Extensions: "permessage-deflate; client_max_window_bits"
 Host: "server-socketio-kevin.herokuapp.com"
```

Figura 9: Ejemplo de conexión WebSocket.
Elaborado por: Kevin Changoluisa, Bryan Imbaquingo

Mientras tanto en la figura 10 se presenta la respuesta por parte del servidor a la solicitud del cliente:

Respuesta del servidor WebSocket

```
Response Headers
 Connection: "Upgrade"
 Upgrade: "websocket"
 Sec-Websocket-Accept: "sSdnk1j3WYfvQveBVdzqJm9fFHA="
 Via: "1.1 vegur"
```

Figura 10: Ejemplo de conexión WebSocket.
Elaborado por: Kevin Changoluisa, Bryan Imbaquingo

Una vez que se ha completado la actualización, la conexión WebSocket entre el cliente y el servidor se establece utilizando la misma conexión subyacente utilizada durante la fase inicial de la comunicación (intercambio de señales) y puede comenzar la comunicación en ambas direcciones, tal como se muestra en la figura 11.

Comunicación WebSocket

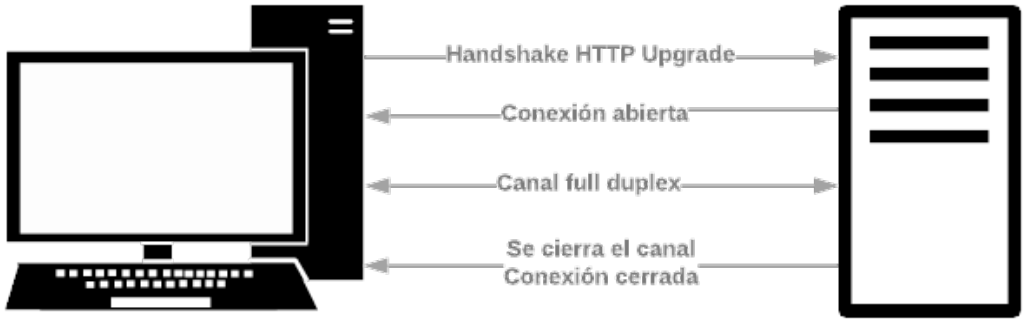


Figura 11: Comunicación WebSocket cliente-servidor.

Elaborado por: Kevin Changoluisa, Bryan Imbaquingo

1.6.2 Ventajas de utilizar WebSocket

- Los WebSockets son más eficientes y de mayor rendimiento que otras soluciones.
- Requieren menos ancho de banda y reducen la latencia.
- Simplifican las arquitecturas de aplicaciones en tiempo real.
- Los WebSockets no requieren encabezado para enviar mensajes, lo que reduce el ancho de banda requerido.

Capítulo 2

Marco metodológico

En este apartado se explica los aspectos principales que corresponden a la aplicación web, la programación, los métodos, las configuraciones y el funcionamiento correcto.

2.1 Arquitectura cliente-servidor

La arquitectura implementada en la aplicación web es cliente-servidor, el servidor establecerá una canal de comunicación WebSocket por el cual se transmitirá la información meteorológica cada vez que el cliente acceda a la aplicación web, o cada vez que se genere un nuevo archivo meteorológico procesado, el cual será enviado desde el servidor FTP hacia el servidor Web. En la figura 12 se puede observar el ciclo de vida de la aplicación.

Ciclo de vida de la aplicación

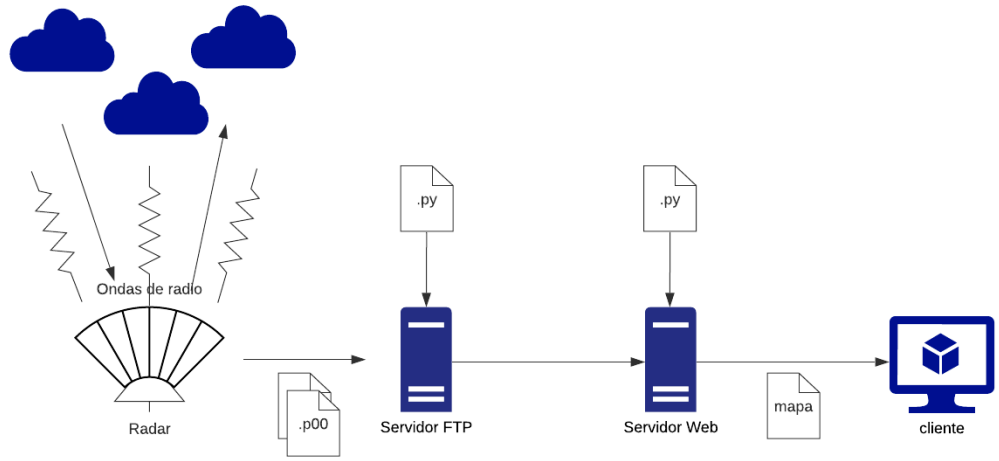


Figura 12: Ciclo de vida de la app
Elaborado por Kevin Changoluisa y Bryan Imbaquingo

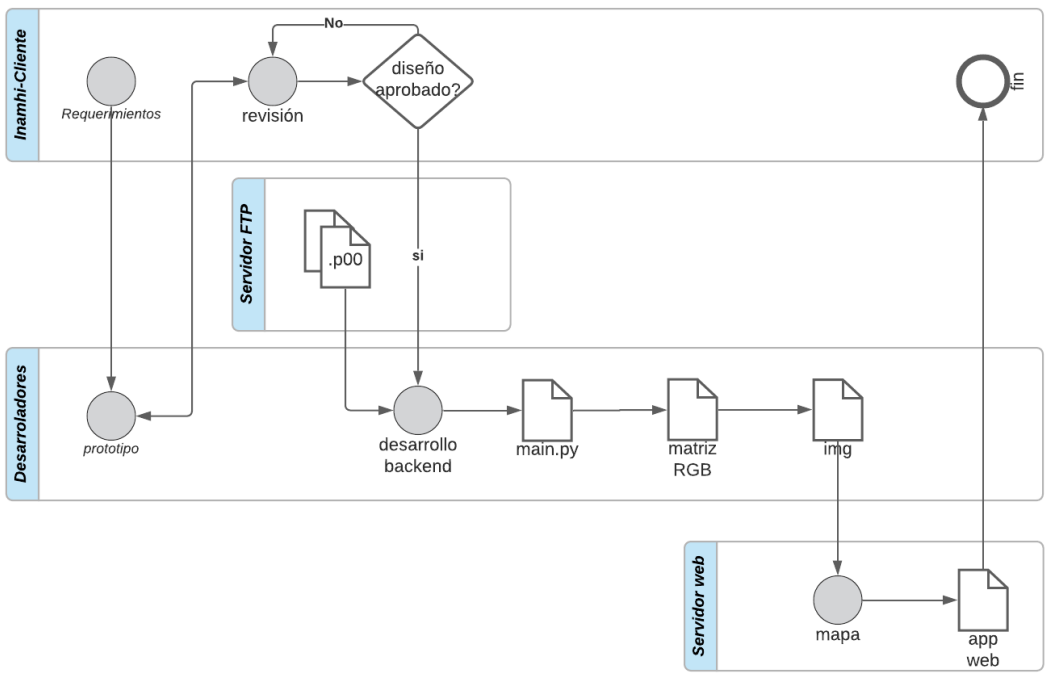


Figura 13: Flujo de desarrollo
Elaborado por Kevin Changoluisa y Bryan Imbaquingo

2.2 Análisis de Requerimientos

Los requerimientos se dividen en funcionales (RF) y no funcionales (RNF). Los RF son aquellos que describen el comportamiento o función particular del sistema o software y los RNF son requerimientos que tratan características generales del sistema como: eficiencia, seguridad, velocidad, etc. Estos evalúan las especificaciones del sistema. En la Tabla 1 se muestran los requerimientos funcionales, mientras que en la Tabla 2 se detallan los requerimientos no funcionales.

Tabla 1: Requerimientos Funcionales

Código	**Descripción**
RF1	Consultar el clima actual en Quito
RF2	Procesar los archivos (p00) en Python, enviados por el radar al servidor del INAMHI, aplicando el filtro de la mediana
RF3	Crear una matriz de pixeles a partir de los archivos procesados para que representen las precipitaciones sobre Quito
RF4	Presentar de manera gráfica la información de los archivos procesados sobre un mapa para definir la trayectoria de las precipitaciones
RF5	Crear un backend en Python que se conecta al servidor del INAMHI y cada vez que llegue un nuevo archivo p00 este sea procesado y posteriormente mostrado en el mapa

Tabla 2: Requerimientos No Funcionales

Código	**Descripción**
RNF1	Desarrollo de una interfaz intuitiva para el usuario
RNF2	Implementación de arquitectura MVC (modelo, vista, controlador) para manejar la arquitectura del sistema
RNF3	Manejo de librerías de python para: aplicar filtros, tratamiento de los P00 y emitir las imágenes
RNF4	La aplicación debe ser escalable y mantenible con el tiempo para que pueda brindar un buen servicio para el INAMHI
RNF5	El sistema no debe ocupar mucha memoria en el servidor del INAMHI
RNF6	Debe tener una disponibilidad permanente para mejorar las observaciones meteorológicas
RNF7	Cualquier usuario con conexión a internet puede ingresar al sitio web
RNF8	El sistema debe ser compatible su funcionamiento con navegadores actuales como: * Google Chrome * Firefox * Brave * Microsoft Edge
RNF9	Desarrollar un backend utilizando python con librerías OpenSource

2.3 Historias de usuario

Tabla 3: Historia de usuario 1

Código Requerimiento	**RF1**
Actor	Usuario
Flujo normal	El usuario accede al portal web La aplicación web carga los datos en la pagina La aplicación muestra el clima actual en un mapa sobre la ciudad de Quito.
Flujo Alternativo	La aplicación no se puede conectar al servidor La aplicación no muestra signos de precipitación en el mapa debido a que está despejado el cielo

2.4 Características de hardware y software

Para aumentar el desempeño de la aplicación web se ha considerado las características de hardware de los equipos del INAMHI 4 y la implementación de nuevas tecnologías de código abierto que permitan un óptimo desempeño de la aplicación.

Tabla 4: Características de hardware

Tipo	Modelo	Procesador	Almacenamiento	Mermoria RAM	Sistema operativo	Función
PC	Dell dl-8000	Intel Core i7 3.4 GHz	2 Tb HDD	8 Gb	Centos 6	Servidor FTP INAMHI
PC	Dell dl-8000	Intel Core i7 3.4 GHz	2 Tb HDD	8 Gb	Windows Server 2012	Servidor Web INAMHI

Tabla 5: Herramientas de desarrollo

Herramienta	Tipo de herramienta	Función
Visual Studio Code	Entorno de desarrollo IDE	Editor de código fuente
Github	Plataforma de desarrollo colaborativo	Repositorio del proyecto
Spyder	Entorno de desarrollo científico IDE	Herramienta para exploración profunda de datos.

2.5 Desarrollo Backend

2.5.1 Construcción Api gestor de procesos del servidor FTP

Para gestionar los eventos que ocurre en el servidor FTP, que almacena los archivos del radar, se ha generado una Api escrita en Python encargada de monitorear, procesar y enviar la información del radar al WebSocket service. En la figura 14 se presenta la secuencia de pasos del Api ejecutado en el servidor de almacenamiento de archivos del radar

A continuación se describe los diferentes procesos que se gestionan en el servidor de almacenamiento de archivos del radar.

2.5.1.1 Monitoreo de directorios y archivos

La fase de monitoreo se encarga de detectar los archivos que envía el radar al servidor FTP. Para monitorear los eventos que ocurre en el servidor en tiempo real se ha utilizado la librería Watchdog de Python. En el momento que Watchdog detecta la creación de nuevos archivos ejecuta algunas operaciones para validar que la extensión del archivo sea .p00 (extensión de archivo binario del radar) y extraer su información. En la figura 15 se presenta la secuencia de pasos que se ejecuta en la fase de monitoreo de archivos y en la figura 16 su respectivo código.

Diagrama de flujo para el algoritmo de gestión de procesos del servidor FTP

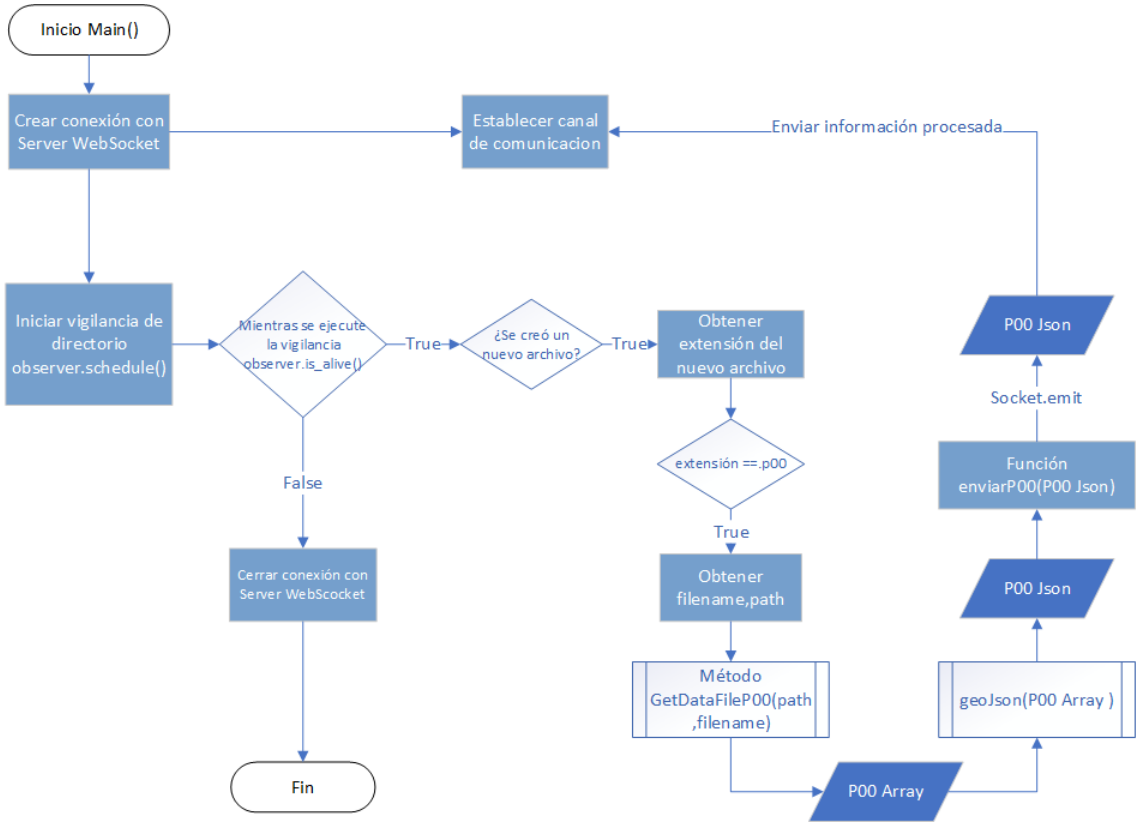


Figura 14: Secuencia de pasos para el algoritmo de gestión de procesos del servidor FTP del INAMHI .

Elaborado por: Kevin Changoluisa, Bryan Imbaquingo

Diagrama de flujo para el algoritmo de monitoreo para directorios y ficheros

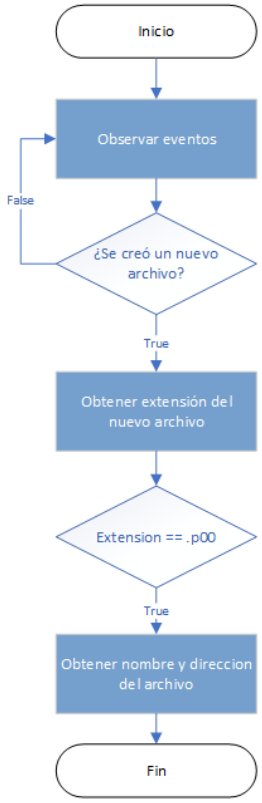


Figura 15: Secuencia de pasos para monitorear directorios y validar ficheros.
Elaborado por: Kevin Changoluisa, Bryan Imbaquingo

Código para monitoreo de directorios y ficheros

```
from watchdog.observers import Observer
from watchdog.events import FileSystemEventHandler
from pathlib import Path

class MyEventHandler(FileSystemEventHandler):
    def on_created(self, event):
        if event.event_type=="created":
            ext=Path(event.src_path).suffix
            if ext=='.p00':
                filename = Path(event.src_path).name
                path=Path(event.src_path).parent.resolve()


observer = Observer()
observer.schedule(MyEventHandler(), ".", recursive=True)
observer.start()
try:
    while observer.is_alive():
        observer.join(1)
except:
    observer.stop()
```

Figura 16: Código para el monitoreo de nuevos archivos meteorológicos.
Elaborado por: Kevin Changoluisa, Bryan Imbaquingo

2.5.1.2 Lectura y procesamiento de archivos de radar

La fase de lectura y procesamiento de archivos, se encarga de decodificar la información de los archivos .p00, almacenarla en una matriz y eliminar el ruido procedente de la exploración del radar. En la figura 17 se presenta la secuencia de pasos que se ejecuta en la fase de lectura y procesamiento, además de su respectivo código en la figura 20.

Diagrama de flujo para el algoritmo de lectura y procesamiento de archivos

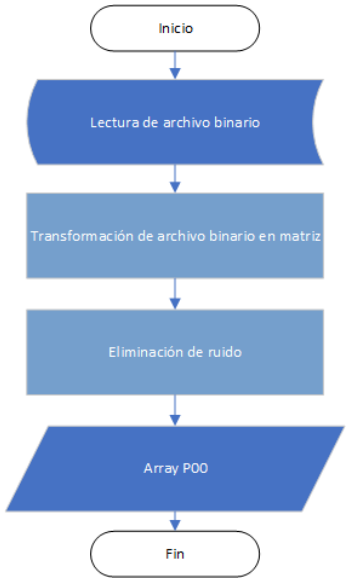


Figura 17: Secuencia de pasos lectura y procesamiento de archivos.
Elaborado por: Kevin Changoluisa, Bryan Imbaquingo

- Lectura de archivos

  Para la lectura de los archivos .p00 se hace uso de la librería **Pathlib** de Python. Pathlib cuenta con muchos métodos para manejo de archivos, el método utilizado para leer y extraer la información contenida de los archivos binario del rada es **read_bytes()**. La información del radar está compuesta por un encabezado del cual obtendremos la fecha y hora de la captura de datos y por los valores de precipitación acumulada.

- Transformación de archivos

  Los valores de precipitación del archivo binario serán almacenado en una matriz de di-

mensión 1x57600 para posteriormente transformarle a una matriz de 240x240. Para este proceso se utilizó la librería **Numpy** de Python, que es ideal para la manipulación de matrices por su rapidez de procesamiento. Mediante el método **reshape()** de Numpy se obtiene la matriz de 240x240 seccionando la matriz de dimensiones 1x57600 en secciones de 240 elementos.

En la figura 18 se presenta como trabaja el método reshape en las matrices.

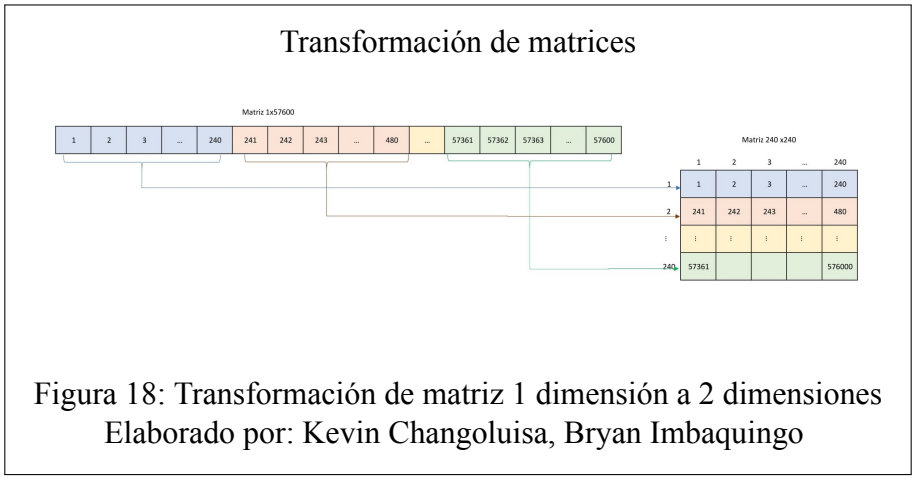


Figura 18: Transformación de matriz 1 dimensión a 2 dimensiones
Elaborado por: Kevin Changoluisa, Bryan Imbaquingo

- Eliminación de ruido en imagen meteorológica

  La matriz de dimensión 240x240 almacena la información de la imagen meteorológica y cada elemento equivale a un pixel con su respectivo valor de precipitación, en varias observaciones como los de la figura 19, se comprobó la existencia de ruido en las imágenes, de tipo sal y pimienta, para reducir el ruido en la imagen se utilizó el filtrado de la mediana con kernel de 3.

  Para reducir el ruido presente en la imagen se utilizó el método **medianBlur()** de la librería **OpenCV** especializada en el procesamiento de imágenes.

Ruido sal y pimienta imágenes meteorológicas

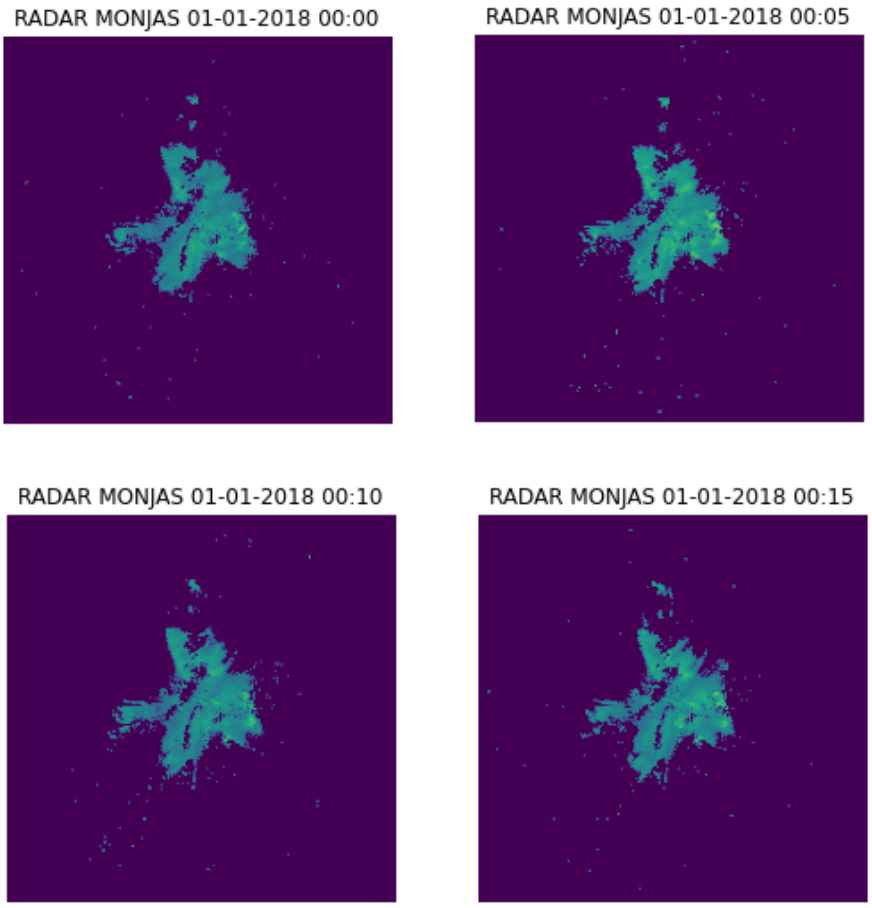


Figura 19: Imágenes meteorológicas radar Furuno sin filtro
Elaborado por: Kevin Changoluisa, Bryan Imbaquingo

Código de lectura y procesamiento de archivos de radar

```
import numpy as np
import cv2
from pathlib import Path

bins = 240
data = Path(str(path)+'\\'+filename).read_bytes()
dateFile=(data[17:29].decode("utf-8")).strip()
dateFile=''+dateFile[6:8]+'/'+dateFile[4:6]+'/'+dateFile[0:4]+''
         +dateFile[8:10]+':'+dateFile[10:14]
vect_Data=np.zeros((1,(bins*bins)), dtype=np.uint8)

for i in range(0,(bins*bins)):
    vect_Data[0][i]=data[i+99]

ArrayP00=np.array(vect_Data).reshape(bins,bins)
arrayP00FilterMedian=cv2.medianBlur(ArrayP00,3)
```

Figura 20: Código de lectura y procesamiento de archivos binarios del radar Furuno.

Elaborado por: Kevin Changoluisa, Bryan Imbaquingo

2.5.1.3 Georreferenciación de los datos

La georreferenciación es el método que nos permite determinar la posición de los elementos de una matriz en un sistema de coordenadas de mapa, como se muestra en la figura 21.

Sistema de coordenadas

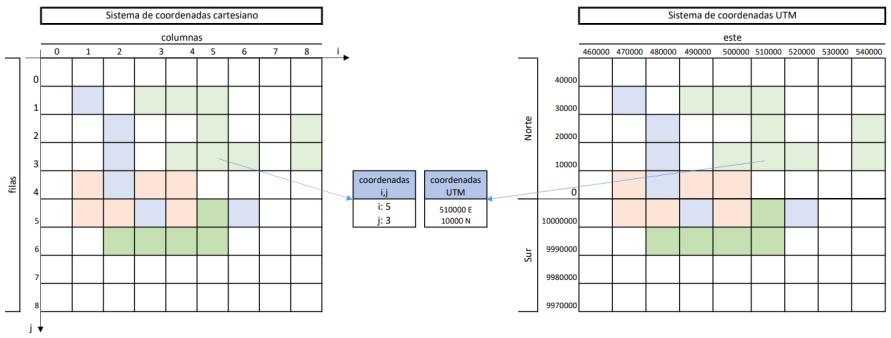


Figura 21: Representación gráfica de los sistemas de coordenadas cartesiano y UTM estándar

Elaborado por: Kevin Changoluisa, Bryan Imbaquingo

En este proceso obtendremos la ubicación geográfica (latitud y longitud) de cada elemento de la matriz procesada, partiendo de la ubicación geográfica del radar en dirección norte, una distancia igual a su radio, después el desplazamiento será hacia el oeste, una distancia igual a su radio. Esta traslación nos aproxima a la posición del primer elemento de la matriz, además se debe tomar en cuenta el cambio de hemisferio al pasar por la línea del Ecuador y el cambio de zonas UTM para realizar la conversión de coordenadas de UTM a latitud y longitud. En la figura 22 se representa la traslación y ubicación de los elementos de la matriz en el plano.

Georreferenciación de datos meteorológicos

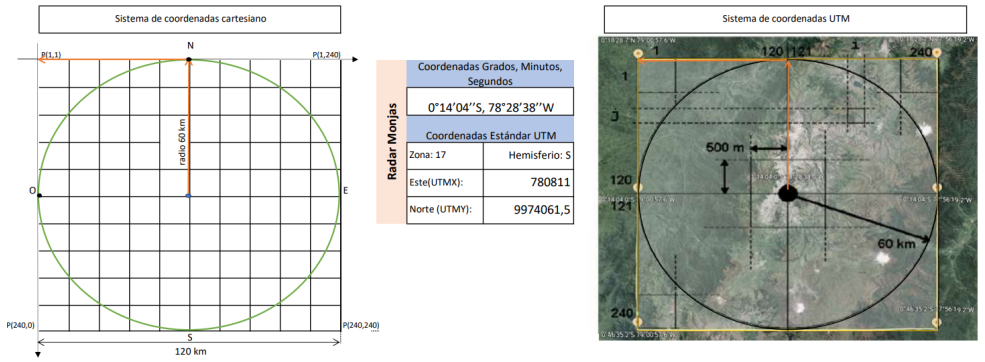


Figura 22: Representación gráfica de la georreferenciación de coordenadas cartesianas a UTM estándar.

Elaborado por: Kevin Changoluisa, Bryan Imbaquingo

A continuación, se presenta en la figura 23 la secuencia de pasos que se ejecuta en la fase de georreferenciación de los datos meteorológicos, además de su respectivo código en la figura 24.

Diagrama de flujo para el algoritmo de lectura y procesamiento de archivos

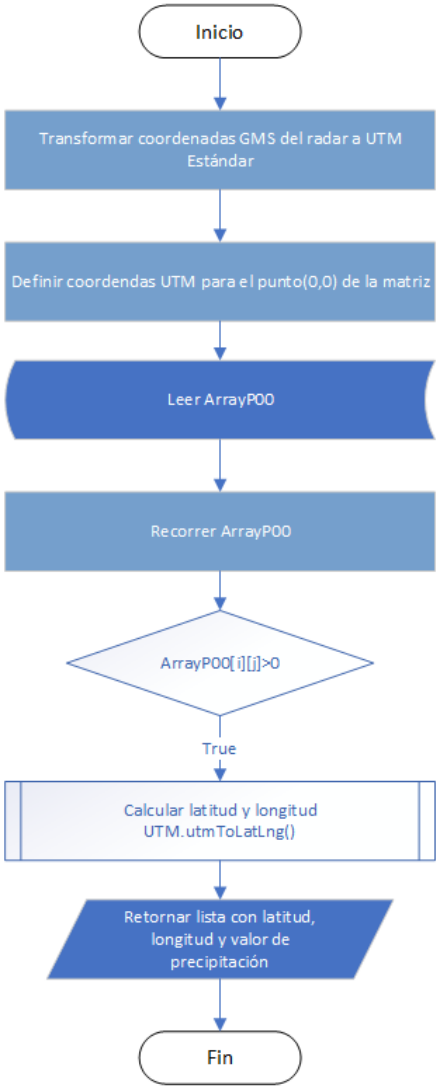


Figura 23: Secuencias de pasos del algoritmo de lectura y procesamiento de archivos del radar Furuno.

Elaborado por: Kevin Changoluisa, Bryan Imbaquingo

Código de lectura y procesamiento de archivos de radar

```
import UTMtolt_lng as UTM

MatP00
vPixel=500                  #valor de cada pixel en metros
origenRadarE=780811         #Punto de origen del radar en UTM
origenRadarN=9974061.5      #Punto de origen del radar en UTM
vradioMetros=60000          #Valor del radio del radar
limite=10000000             #Limite hemisferio norte-sur
hemNorte=True
zona=17                     #Zona Utm donde opera el radar

PxOrigen=origenRadarE-(vradioMetros+vPixel)
PyOrigen=origenRadarN+(vradioMetros+vPixel)-limite

Lt_Lg_Valor={"data":[]}


for i in range(MatP00.shape[0]):
    PxNewOrigen=PxOrigen
    if PyOrigen>0:
        PyOrigen=PyOrigen-vPixel
    elif PyOrigen<=0:
        hemNorte=False
        PyOrigen=limite

    for j in range(MatP00.shape[1]):
        if PxNewOrigen<833979:
            zona=17
        else:
            zona=18
        if MatP00[i][j] >0:
            lat,lng=UTM.utmToLatLng(zona, PxNewOrigen, PyOrigen,hemNorte)
            reflec=int(MatP00[i][j])
            Lt_Lg_Valor["data"].append(
               {"mag":reflec,
                "coordinates": [lat,lng]})
        PxNewOrigen=PxNewOrigen+vPixel
```

Figura 24: Código de lectura y procesamiento de archivos.
Elaborado por: Kevin Changoluisa, Bryan Imbaquingo

2.5.1.4 Comunicación con WebSocket service

La conexión con el WebSocket service se establece como primer paso en la ejecución del programa, creando un canal de comunicación siempre abierto para transmitir los datos meteorológicos en tiempo real. Para este proceso se ha utilizado la librería Socket.IO de Python garantizando la compatibilidad con las demás versiones de los protocolos Socket.IO y su reconexión automática ante cualquier falla del servidor WebSocket.

A continuación, se presenta en la figura 25 la secuencia de pasos que se ejecuta en la fase de georreferenciación de los datos meteorológicos, además de su respectivo código en la figura 26.

Diagrama de flujo para el algoritmo de comunicación con servidor

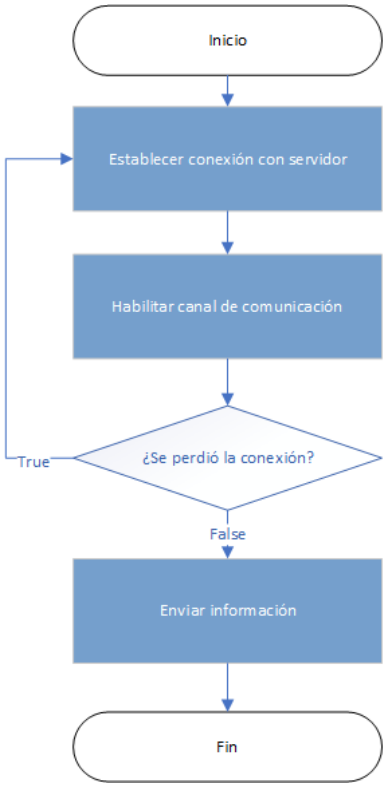


Figura 25: Secuencia de pasos para establecer la comunicación con el servidor y envío de datos meteorológicos.

Elaborado por: Kevin Changoluisa, Bryan Imbaquingo

Código para comunicación con servidor

```
import socketio
sio = socketio.Client(reconnection=True)
sio.connect('wss://server-socketio.com/')
@sio.event
def enviarP00(data):
    sio.emit('obtenerP00', data)
```

Figura 26: Código para establecer la comunicación con el servidor y envío de datos meteorológicos

Elaborado por: Kevin Changoluisa, Bryan Imbaquingo

2.5.2 Construcción Api WebSocket service para transferencia de datos

WebSocket service es el encargado de gestionar las conexiones y establecer los métodos para la recepción y transmisión de datos. El Api se desarrolló en Python con el micro-framework Flask y la librería SocketIO. A continuación, se presenta en la figura 27 la secuencia de pasos que se ejecuta en la fase de georreferenciación de los datos meteorológicos, además de su respectivo código en la figura 28.

Diagrama de flujo para el manejo y transferencia de datos WebSocket service

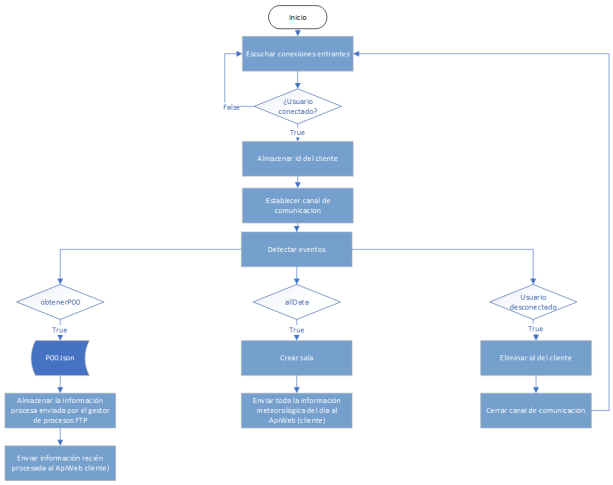


Figura 27: Secuencia de pasos para el manejo y transferencia de datos WebSocket service.

Elaborado por: Kevin Changoluisa, Bryan Imbaquingo

Código para el manejo y transferencia de datos WebSocket service

```
import os
from flask import Flask, request
from flask_cors import CORS
from flask_socketio import SocketIO

app = Flask(__name__)
app.config['SECRET_KEY'] = os.environ.get('SECRET')
socketio = SocketIO(app, manage_session=False,cors_allowed_origins="*")
CORS(app)

users = []
archivosP00=[]

@socketio.on('connect')
def on_connect():
    print('Client connected')
    users.append(request.sid)
    print("Total de usarios conectados: ", len(users))

@socketio.on('disconnect')
def on_disconnect():
    print('Client disconnect')
    users.remove(request.sid)
    print("Total de usarios conectados: ", len(users))

@socketio.on('allData')
def getAllDataP00():
    print("Send all Data")
    socketio.emit('getAllDataP00',archivosP00,room=request.sid)

@socketio.on('obtenerP00')
def setListP00(data):
    print("Recibiendo")
    archivosP00.append(data)
    socketio.emit('getDataP00',data)

if __name__ == '__main__':
    print("Iniciando")
    app.run(debug=True)
```

Figura 28: Código para el manejo y transferencia de datos WebSocket service.
Elaborado por: Kevin Changoluisa, Bryan Imbaquingo

2.6 Desarrollo Frontend

En la figura 29 se observa el diagrama de flujo que describe la secuencia de pasos que suceden en el frontend de la aplicación web.

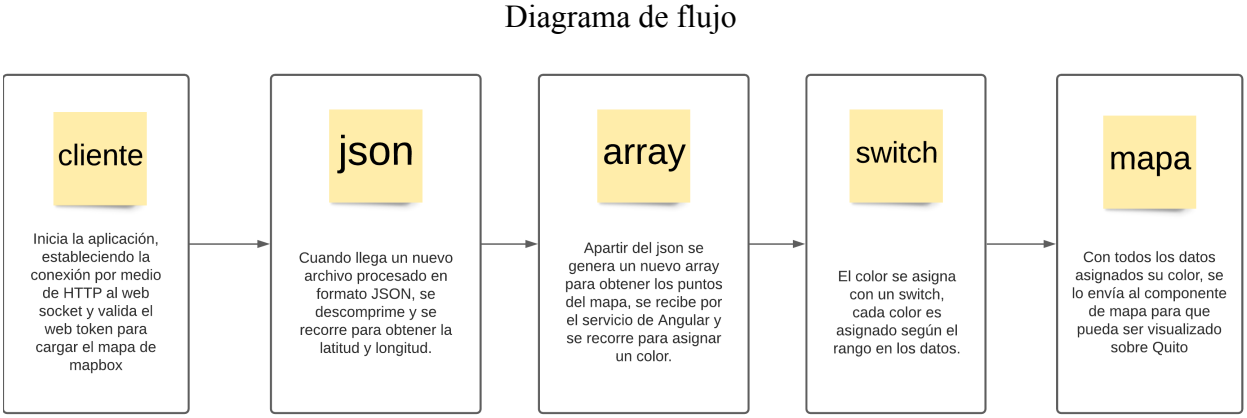


Figura 29: Diagrama de pasos desde el cliente (frontend)

Elaborado por: Kevin Changoluisa, Bryan Imbaquingo

2.6.1 Diseño de interfaz gráfica

Para el desarrollo del interfaz gráfica de la aplicación web se utilizó el framework Angular, basado en el diseño Modelo Vista Controlador (MVC) que permite reutilizar cada elemento existente dentro de la aplicación web (componente), en este caso la aplicación cuenta con su componente principal que contiene el mapa del Ecuador, la ubicación del radar Furuno y el radio de cobertura de la información meteorológica enviado desde el backend en formato Json. En la figura 30 se presenta una vista general del prototipo de la aplicación web aprobado por el cliente.

Prototipo de la interfaz gráfica del aplicativo web

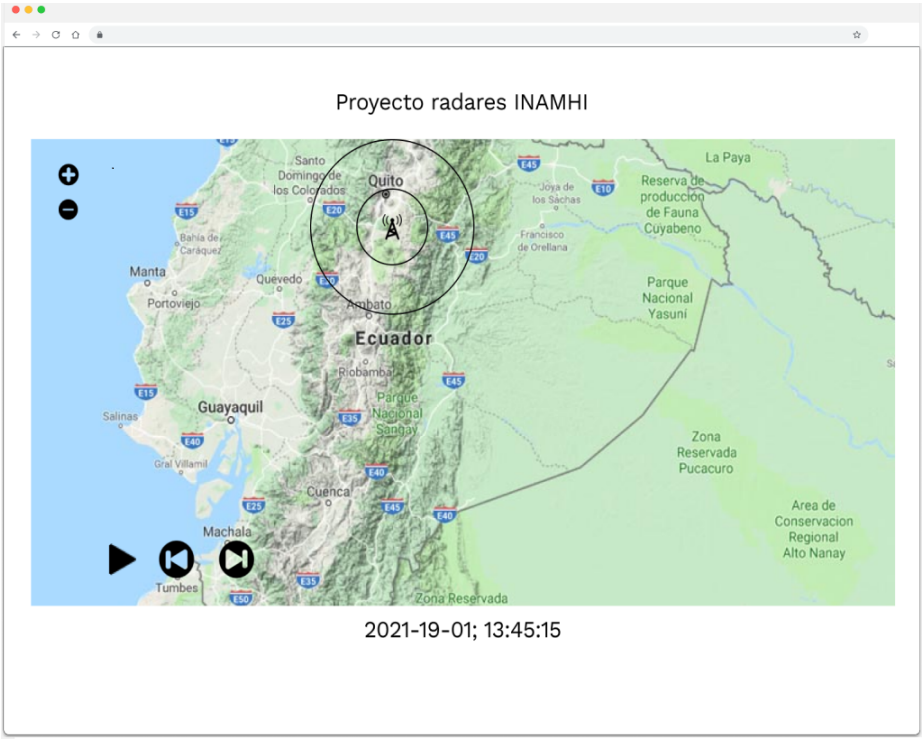


Figura 30: Prototipo de la interfaz gráfica del aplicativo web, usando Figma

Elaborado por: Kevin Changoluisa, Bryan Imbaquingo

2.7 Pruebas

En esta sección se detalla las pruebas que se realizó a la aplicación web. Los resultados van de acuerdo a las características descritas en la tabla 4 Heroku ya que la aplicación actualmente funciona en esta plataforma de servicios en la nube, durante la fase de prueba se evaluará:

- Rendimiento (Pingdom Tools)
- Pruebas de estrés (Apache JMeter)
- Prueba de carga (Script Python)
- Compatibilidad. (LambdaTest)

2.7.1 Rendimiento (Pingdom Tools)

Es una herramienta on-line que permite analizar el rendimiento y las interacciones de un sitio web para mejorar la experiencia de usuario.

Resultados Pingdom Tools

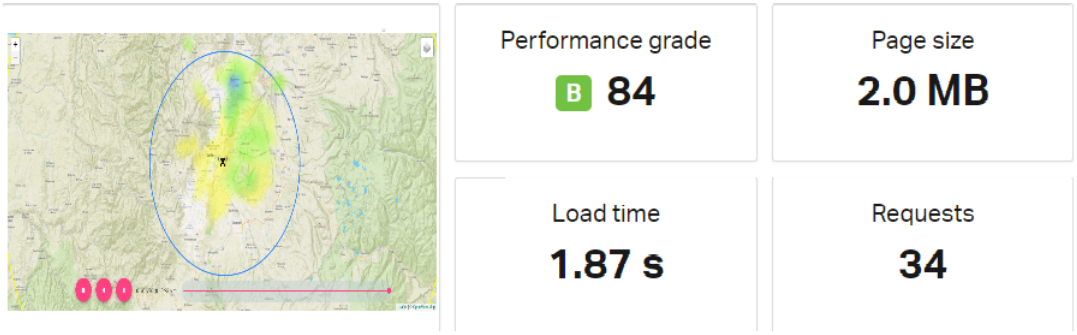


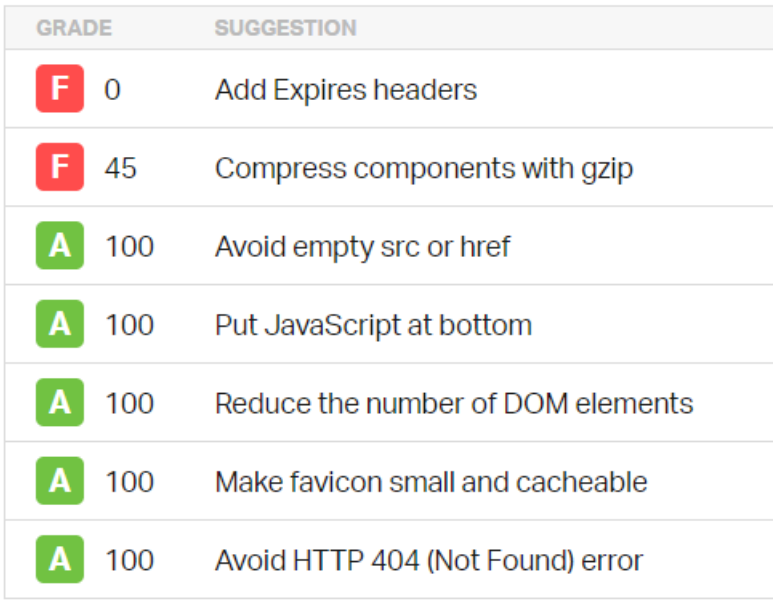
Figura 31: Resultados obtenidos de Pingdom Tools

Elaborado por: Kevin Changoluisa, Bryan Imbaquingo

Los resultados obtenidos que se pueden apreciar en la figura 31, indica que se realizó 34 peticiones al sitio web, el rendimiento es 84/100, esto quiere decir que el sitio tiene un corto tiempo de carga, es fácil de usar. La página carga aproximadamente en 1.87 segundos, esto indica que tiene una carga veloz de sus componentes.

Los resultados de la figura 32, indica que la aplicación web es muy buena, salvo por 2 sugerencias que se indica en rojo, la puntuación de 45 sugiere sobre comprimir el sitio para mejorar el tiempo de carga y la puntuación de 0 explica que sería mejor realizar un menor número de peticiones HTTP. Durante las 34 peticiones realizadas de manera automática por la herramienta, nos indica que el sitio tuvo como respuesta ”status 200”, como se ve en la figura 33, esto indica que el acceso al sitio fue un éxito.

Resultados Pingdom Tools

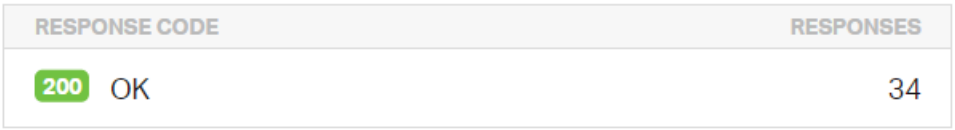


GRADE		SUGGESTION
F	0	Add Expires headers
F	45	Compress components with gzip
A	100	Avoid empty src or href
A	100	Put JavaScript at bottom
A	100	Reduce the number of DOM elements
A	100	Make favicon small and cacheable
A	100	Avoid HTTP 404 (Not Found) error

Figura 32: Resultados obtenidos de Pingdom Tools

Elaborado por: Kevin Changoluisa, Bryan Imbaquingo

Resultados Pingdom Tools - Peticiones



RESPONSE CODE		RESPONSES
200	OK	34

Figura 33: Resultados Pingdom Tools - Peticiones

Elaborado por: Kevin Changoluisa, Bryan Imbaquingo

2.7.2 Prueba de estrés (JMeter)

Para determinar la solidez de la aplicación en los momentos de carga extrema se ha utilizado el software JMeter para realizar conexiones simultáneas entre el servidor WebSocket y los clientes. Para esta prueba someteremos al servidor a 100,1000 y 10000 conexiones simultáneas.

2.7.2.1 Prueba con 100 usuarios en JMeter

En la gráfica 34 se muestran los resultados de las 100 conexiones realizadas al servidor. Todas las conexiones fueron establecidas exitosamente.

Resultados prueba de estrés 100 usuarios

Label	#Samples	Average	Min	Max	Srd. Dev.	Error %	Throughput	Received Kb/sec	Sent Kb/sec	Avg. Bytes
API SERVER WEB SOCKET: Open Connection WEB SOCKET SERVER	100	173	171	243	7,41	0,00%	9,9/sec	1,41	2,31	145,0

Figura 34: Prueba de estrés JMeter 100 usuarios
Elaborado por: Kevin Changoluisa, Bryan Imbaquingo

2.7.2.2 Prueba con 1000 usuarios en JMeter

En la gráfica 35 se muestran los resultados de las 1000 conexiones realizadas al servidor. Todas las conexiones fueron establecidas exitosamente.

Resultados prueba de estrés 1000 usuarios

Label	#Samples	Average	Min	Max	Srd. Dev.	Error %	Throughput	Received Kb/sec	Sent Kb/sec	Avg. Bytes
API SERVER WEB SOCKET: Open Connection WEB SOCKET SERVER	1000	172	170	247	4,3	0,00%	98,4/sec	13,93	22,87	145,0

Figura 35: Prueba de estrés JMeter 1000 usuarios
Elaborado por: Kevin Changoluisa, Bryan Imbaquingo

2.7.2.3 Prueba con 10000 usuarios en JMeter

En la gráfica 36 se muestran los resultados de las 10000 conexiones realizadas al servidor. Aquí encontramos que no se pueden establecer con normalidad 302 conexiones.

Resultados prueba de estrés 10000 usuarios

Label	#Samples	Average	Min	Max	Srd. Dev.	Error %	Throughput	Received Kb/sec	Sent Kb/sec	Avg. Bytes
API SERVER WEB SOCKET: Open Connection WEB SOCKET SERVER	10000	1118	170	20015	3454,95	3,02%	33,4/sec	45,78	75,15	140,6

Figura 36: Prueba de estrés JMeter 10000 usuarios
Elaborado por: Kevin Changoluisa, Bryan Imbaquingo

2.7.3 Prueba de carga

Se ha desarrollado un programa en Python que simulará él envió de información del radar al servidor FTP, la información que será enviada corresponde a un día normal de funcionamiento del radar, el tiempo de espera para el envió de archivos será 1 segundo.

Como se puede visualizar en la figura 37 ,los resultados para esta prueba son 288 archivos P00 procesados correctamente con una media de 0.19 segundos en procesar la información del radar, además toda la información fue enviada correctamente al servidor web.

Detalle de resultados prueba de carga

	Hora	Nombre	Tiempo de procesamiento	Estado
1	18:38:30	mnjs0010.p00	0.1290 s	enviado
2	18:38:31	mnjs0015.p00	0.1290 s	enviado
3	18:38:32	mnjs0020.p00	0.1310 s	enviado
4	18:38:33	mnjs0025.p00	0.1270 s	enviado
5	18:38:34	mnjs0030.p00	0.1250 s	enviado
6	18:38:35	mnjs0035.p00	0.1280 s	enviado
7	18:38:36	mnjs0040.p00	0.1260 s	enviado
8	18:38:37	mnjs0045.p00	0.1280 s	enviado
9	18:38:38	mnjs0050.p00	0.1280 s	enviado
...	...	...	...	...
280	18:54:56	mnjs2315.p00	0.0900 s	enviado
281	18:54:57	mnjs2320.p00	0.0890 s	enviado
282	18:54:58	mnjs2325.p00	0.0960 s	enviado
283	18:54:59	mnjs2330.p00	0.0900 s	enviado
284	18:55:00	mnjs2335.p00	0.0880 s	enviado
285	18:55:01	mnjs2340.p00	0.0880 s	enviado
286	18:55:02	mnjs2345.p00	0.0890 s	enviado
287	18:55:03	mnjs2350.p00	0.0890 s	enviado
288	18:55:04	mnjs2355.p00	0.0890 s	enviado

Total de archivos procesados:	288
Tiempo medio en procesamiento (s):	0.1918
Total de archivos enviados:	288

Figura 37: Resultados de la prueba de carga
Elaborado por: Kevin Changoluisa, Bryan Imbaquingo

2.7.4 Compatibilidad (LambdaTest)

Para medir la compatibilidad de la aplicación con diferentes navegadores y diferentes sistemas operativos se utilizó la herramienta online LambdaTest, el servicio de Lambdatest permite realizar esta tarea con facilidad. A continuación en la tabla 6 se detallan los resultados obtenidos por esta herramienta.

Tabla 6: Detalle de compatibilidad con diversos navegadores según Lambdatest

SISTEMA OPERATIVO	NAVEGADOR	VERSION	ACCIONES	RESULTADO
macOS Big Sur	Chrome	94	Ingresar a la aplicación	OK
			Cargar componente principal	OK
			Manipular el mapa	OK
			Salir del sistema	OK
Windows 7	Firefox	96	Ingresar a la aplicación	OK
			Cargar componente principal	OK
			Manipular el mapa	OK
			Salir del sistema	OK
macOS Sierra	Safari	10.1	Ingresar a la aplicación	OK
			Cargar componente principal	OK
			Manipular el mapa	OK
			Salir del sistema	OK
Windows 10	Opera	83	Ingresar a la aplicación	OK
			Cargar componente principal	OK
			Manipular el mapa	OK
			Salir del sistema	OK

NOTA: Compatibilidad registrada por Lambdatest en diferentes navegadores

Capítulo 3

Pruebas y Resultados

3.1 Información meteorológica procesada

En la figura 38 se presenta la información meteorológica procedente del radar Furuno, la cual fue procesada para eliminar el ruido de tipo sal y pimienta mediante la aplicación del filtro mediano con un kernel de 3.

3.2 Interfaz gráfica de la aplicación web

La aplicación web cuenta con una sola interfaz gráfica interactiva, capaz de actualizar automáticamente las condiciones climáticas sin necesidad de recargar la página web. Cuando el radar no detecta precipitaciones en el ambiente se visualizará un mapa despejado, como se puede observar en la figura 39. Sin embargo si el radar emite información meteorológica con valores altos de precipitación se procederá a graficar según su escala de colores ubicado al costado izquierdo de la página web, como se puede observar en la figura 40

El código se encuentra disponible en Github en el siguiente enlace: Github: código fuente

Información meteorológica con filtro mediano

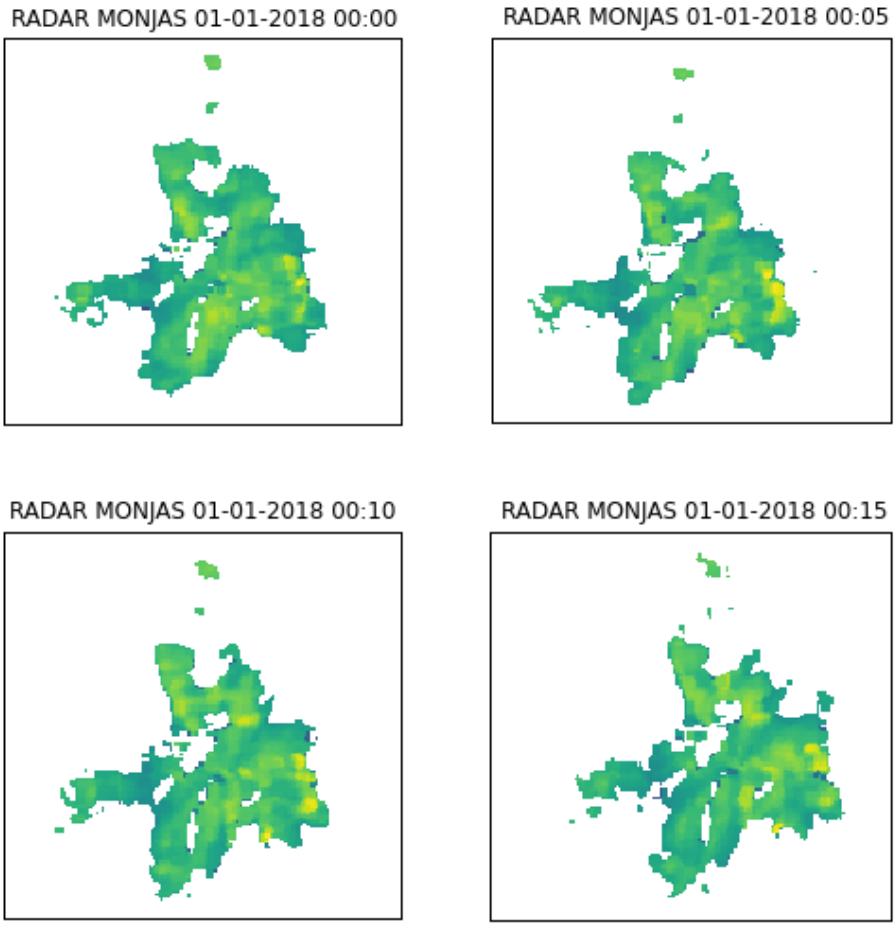


Figura 38: Resultados del procesamiento de imágenes meteorológicas procedentes del radar Furuno en Quito-Ecuador.

Elaborado por: Kevin Changoluisa, Bryan Imbaquingo

Frontend - mapa sin precipitaciones

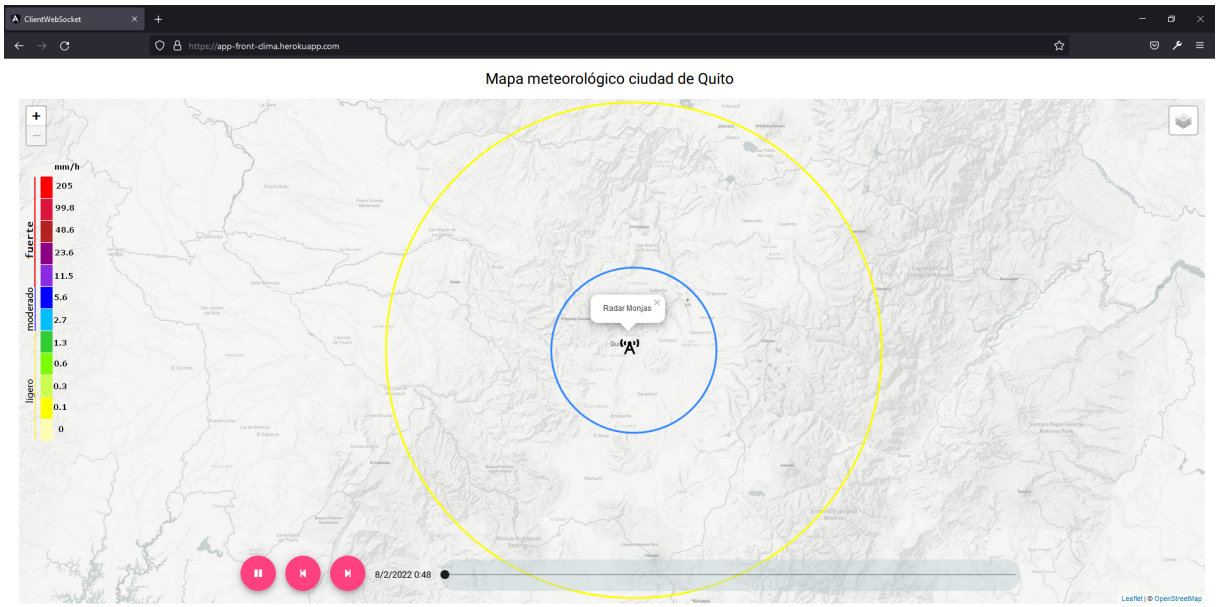


Figura 39: Aplicación web en funcionamiento sin precipitaciones en el mapa.

Elaborado por: Kevin Changoluisa, Bryan Imbaquingo

Frontend - mapa con precipitaciones

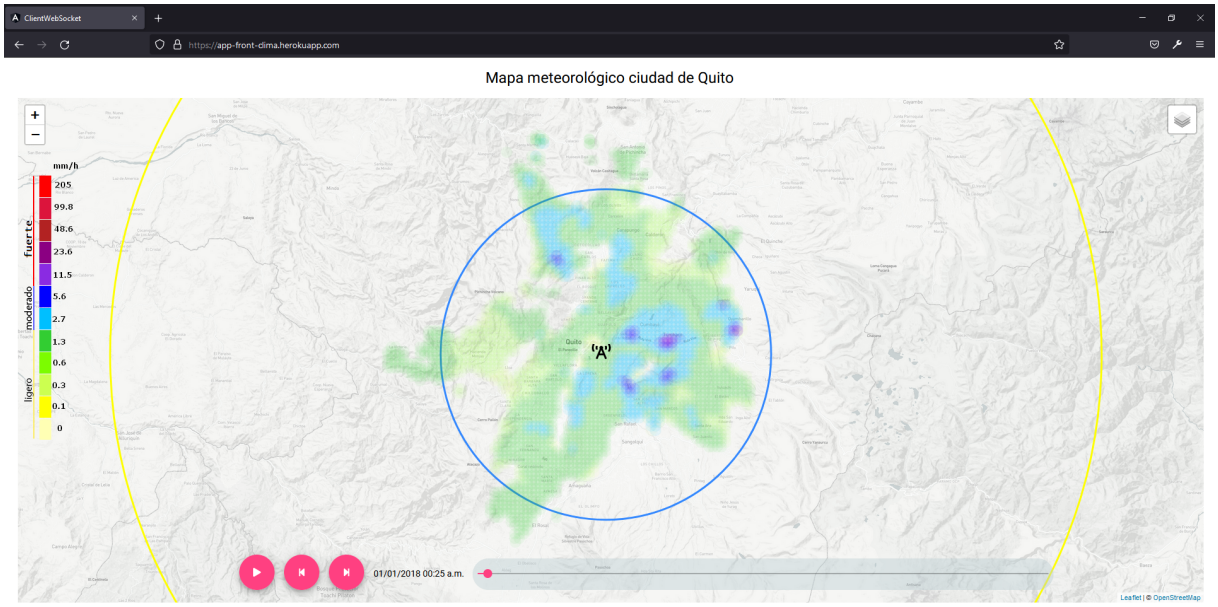


Figura 40: Aplicación web en funcionamiento con precipitaciones en el mapa.

Elaborado por: Kevin Changoluisa, Bryan Imbaquingo

Conclusiones

En este proyecto se desarrolló una aplicación de streaming que permitirá mejorar la transmisión de datos del radar Furuno en la ciudad de Quito. De este modo, se presentan las siguientes conclusiones:

- La aplicación web cumplió con los objetivos y requerimientos acordados con el cliente (INAMHI) en el plazo establecido. Como resultado se obtiene una aplicación web con tecnologías actuales y compatibles con la mayoría de navegadores, encargada de transmitir en tiempo real las condiciones ambientales del distrito metropolitano de Quito. La información transmitida está conformada por imágenes meteorológicas previamente procesadas, para reducir el ruido generado por el radar en el momento de escaneo.

- El desarrollo de herramientas tecnológicas para el monitoreo constante del clima es de suma importancias para emitir alertas tempranas ante posibles catástrofes naturales, la aplicación web se adapta a los tiempos de transmisión de archivos (P00) del radar Furuno, por lo tanto, se puede informar de fenómenos meteorológicos como la lluvia en mejor resolución tiempo-espacio.

- La implementación de la tecnología WebSocket como canal de comunicación entre el cliente y el servidor, ayudó a reducir el tiempo de transmisión de las imágenes meteorológicas previamente procesadas. De tal manera, que la aplicación web podrá presentar las imágenes en un tiempo menor a 1 segundo, desde el momento que llega nueva información del radar al servidor de almacenamiento de archivos P00. La ventaja de usar esta tecnología es que cliente no deberá recargar la página web para poder visualizar la nueva información procesada, ya que al establecer un canal full dúplex los datos se actualizarán automáticamente del lado del cliente.

- La utilización de Python como lenguaje de programación para el análisis y procesamiento de los datos meteorológicos, fue beneficioso gracias al conjunto de librerías orientadas al análisis de datos que simplifican ciertas tareas y reducen tiempo de desarrollo en la creación de código.

- Los conocimientos adquiridos en la formación académica universitaria fueron de mucha utilidad para la construcción de esta aplicación web del INAMHI, ya que fue posible combinarlos con el conocimiento en otras ciencias como la Meteorología, Radioastronomía o Geografía, para lograr dar solución a un problema real

Recomendaciones

Para mejorar la transmisión de información meteorológica se recomienda:

- Se debe Reducir el tiempo de escaneo del radar Furuno, ya que actualmente está configurado para generar imágenes meteorológicas cada cinco minutos, el software que se desarrolló procesa imágenes en un tiempo menor a un segundo. Para aprovechar las capacidades del nuevo software, es necesario reducir el tiempo de escaneo del radar.

- Aumentar el ancho de banda en el servidor web, esto permitirá que no exista retrasos en la transmisión de la información al cliente y va a permitir conexiones simultaneas mayores a diez mil usuarios

- Hacer un mantenimiento continuo y adecuado de las librerías utilizadas en el desarrollo de la aplicación web, ya que son herramientas Open Source que presentan actualizaciones ante posibles fallas.

- Comparar las mediciones del radar con los valores medidos por los pluviómetros instalados en las diferentes estaciones que están dentro del área de cobertura del radar. Esta comparación permitirá determinar la relación existente entre valores medidos del radar y el pluviómetro en un área específica.

Anexos

Repositorio de código fuente

Código fuente

Resultados obtenidos por Lambdatest

Screenshots Lambdatest

Quito 25 de febrero e 2022.

Carta de Aceptación de Proyecto Técnico

De: Darwin Rosero Vaca.

Para: Universidad Politécnica Salesiana.

Durante el periodo comprendido entre el mes de octubre del año 2021 hasta febrero del año 2022, los estudiantes de la Universidad Politécnica Salesiana, de la carrera de Ingeniería en Ciencias de la Computación, **Kevin Andrés Changoluisa Cuayal** con cedula de identificación nro. 1718123563 y **Bryan Andrés Imbaquingo Almagro** con cedula de identificación nro. 17220457330. Desarrollaron el proyecto técnico para su titulación denominado **"Desarrollo de una aplicación de streaming para el envío de imágenes meteorológicas desde un radar Furuno manejado por el INAMHI"**; El cual ha sido terminado con satisfacción según los requerimientos establecidos.

Con estos antecedentes me complace informar mediante esta carta, que el proyecto antes mencionado se acepta con satisfacción, y será una herramienta útil en la Dirección de Información Hidrometeorológica del Instituto nacional de Meteorología e Hidrología (INAMHI).

Atentamente,

Firmado electrónicamente por:
DARWIN ANDRES ROSERO VACA

Darwin Rosero Vaca
Coordinador
Dirección de información Hidrometeorológica

Referencias

(s.f.).

Alegre Gutiérrez, E., Pajares Martinsanz, G., y de la Escalera Hueso, A. (2016). *Conceptos y métodos en visión por computador*. España: Grupo de Visión del Comité Español de Automática (CEA). Descargado de https://intranet.ceautomatica.es/sites/default/files/upload/8/files/ConceptosyMetodosenVxC.pdf

América latina y el caribe: la segunda región más propensa a los desastres | noticias onu. (2020). Organización de las Naciones Unidas (ONU). Descargado de https://news.un.org/es/story/2020/01/1467501

Andrades Rodríguez, M. S., y Múñez León, C. (2012). *Fundamentos de climatología*. Logroño: Universidad de La Rioja. Descargado de https://dialnet.unirioja.es/descarga/libro/267903.pdf

da Silva, E. A., y Mendonça, G. V. (2005). 4 - digital image processing. En W.-K. CHEN (Ed.), *The electrical engineering handbook* (p. 891-910). Burlington: Academic Press. Descargado de https://www.sciencedirect.com/science/article/pii/B9780121709600500645 doi: https://doi.org/10.1016/B978-012170960-0/50064-5

Estrategia para la implantación de software libre en la administración pública central (Inf. Téc.). (2009). Descargado de https://cti.gobiernoelectronico.gob.ec/ayuda/manual/decreto_1014.pdf

Fenton, S. (2017). *Pro typescript : application-scale javascript development*. Berkeley, CA: Apress. doi: 1007/978-1-4842-3249-1

Fortelli, A. (2021). *Elementi di meteorologia per la progettazione green*. FedOA-Federico II University Press. Descargado de http://www.fedoabooks.unina.it/index.php/

fedoapress/catalog/book/287 doi: 10.6093/978-88-6887-108-6

Geo ecuador 2008 : informe sobre el estado del medio ambiente. (2008). Quito, Ecuador Ciudad de Panamá, Panama: FLACSO Ecuador Ministerio del Ambiente PNUMA. Descargado de https://biblio.flacsoandes.edu.ec/libros/digital/41444.pdf

Gilibets, L. (2020, Nov). *Qué es kanban y cómo utilizarlo en el desarrollo de proyectos*. Descargado de https://www.iebschool.com/blog/metodologia-kanban-agile-scrum/

Guerra, P. A. (2002, octubre). Tratamiento y procesamiento de una imagen RADARSAT de cartagena (2000) y aplicación de técnicas de fusión de imágenes. *Boletín Científico CIOH*(20), 79–91. Descargado de https://doi.org/10.26640/22159045.111 doi: 10.26640/22159045.111

Han, S., Go, Y., Noh, H., y Song, H. (2019, enero). Cooperative server-client http adaptive streaming system for live video streaming. En *2019 international conference on information networking (icoin)*. IEEE. Descargado de https://doi.org/10.1109/icoin.2019.8718151 doi: 10.1109/icoin.2019.8718151

Informe n°11 de situación época lluviosa ecuador. (2021). https://www.gestionderiesgos.gob.ec/wp-content/uploads/2021/03/Informe-de-Situacion-No-11-Epoca-Lluviosa-desde-el-01022021.pdf. Servicio Nacional de Gestión de Riesgos y Emergencias. ((Accessed on 01/18/2022))

Jorge Roberto Jova Rodríguez, H. D. R., Alberto Bradshaw Gonzalez. (2019). Streaming de archivos multimedia desde bases de datos. *Revista Cubana de Ciencias Informáticas*, *41*(6), 955 - 968. Descargado de http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S2227-18992015000200001&lang=es

Klipp, P. (2014). Getting started with kanban. *Amazon Digital Services*.

Liu, Q., y Sun, X. (2012). Research of web real-time communication based on web socket. *International Journal of Communications, Network and System Sciences*, *05*(12), 797–801. Descargado de https://doi.org/10.4236/ijcns.2012.512083 doi: 10.4236/ijcns.2012.512083

Llugsi Cañar, R. J. (2013). Modelación para la cuantificación espacio temporal de la precipitación para centros poblados en el ecuador caso: Dm de quito. *INAMHI*.

Melnikov, A., y Fette, I. (2011, diciembre). *The WebSocket Protocol* (no 6455). RFC 6455. RFC Editor. Descargado de https://www.rfc-editor.org/info/rfc6455 doi: 10.17487/RFC6455

Meteorología y climatología : unidad didáctica : Semana de la ciencia y la tecnología 2004. (2004). Madrid: Fundación Española para la Ciencia y la Tecnología. Descargado de https://www.fecyt.es/es/publicacion/unidad-didactica-meteorologia-y-climatologia

Mihaela Juganaru, M. (2014). *Introducción a la programación*. Ciudad de México: Larousse - Grupo Editorial Patria. Descargado de https://editorialpatria.com.mx/pdffiles/9786074384154.pdf

Monsálvez, J. C. G. (2017, agosto). Python como primer lenguaje de programación textual en la enseñanza secundaria. *Education in the Knowledge Society (EKS)*, *18*(2), 147–162. Descargado de https://doi.org/10.14201/eks2017182147162 doi: 10.14201/eks2017182147162

Nachtegael, M., Van der Weken, D., Van De Ville, D., y Kerre, E. E. (2003). *Fuzzy filters for image processing*. Berlin New York: Springer.

Niels, J. (2010). Local area weather radar [Manual de software informático]. Denmark: DHI.

Patilla, H. J., Enciso, E. G., Pulache, J. C. J., Rodríguez, J. L. L., Huallanca, E. S., y Conis-

lla, Y. M. (2021). Modelo de gestión de desarrollo de software ágil mediante scrum y kanban sobre la programación extrema. *Revista Ibérica de Sistemas e Tecnologias de Informação*(E43), 450–466.

Pérez, B. M. (2021, mayo). Streaming: ventajas, desafíos y oportunidades de las radiotelevisiones para captar audiencias. *Revista de Ciencias de la Comunicación e Información*, 45–65. Descargado de https://doi.org/10.35742/rcci.2021.26.e85 doi: 10.35742/rcci.2021.26.e85

Quiroga, J. M. (2018). Primeros desarrollos de tecnología radar en los principales beligerantes de la ii guerra mundial. un análisis desde la perspectiva ciencia, tecnología y sociedad. *Ciencia, docencia y tecnología*, *29*(57), 36–59.

Sanfuentes, J. (2000). Historia del radar. *Revista de Marina. Armada de Chile. Santiago*, 1–7.

Sarría, F. A. (2006). Sistemas de información geográfica. Descargado de https://www.um.es/geograf/sigmur/sigpdf/temario.pdf

Stallman, R. (2004). *Software libre para una sociedad libre*. Madrid: Traficantes de Sueños. Descargado de https://www.gnu.org/philosophy/fsfs/free_software.es.pdf

Tonato, D. (2021). *Propuesta de sostenibilidad al proyecto radares meteorológicos del dmq* (Inf. Téc.). Instituto Nacional de Meteorología e Hidrología (INAMHI).

Wang, L., Zhang, L., y Ma, Y. (2017). Gstreamer accomplish video capture and coding with pygi in python language. En *2017 first international conference on electronics instrumentation information systems (eiis)* (p. 1-4). doi: 10.1109/EIIS.2017.8298579

Wohlgethan, E. (2018). *Supportingweb development decisions by comparing three major javascript frameworks: Angular, react and vue. js* (Tesis Doctoral no publicada). Hochschule für Angewandte Wissenschaften Hamburg.