Escuela de Ingeniería Informática

Universidad de Oviedo

Máster Universitario en Ingeniería Web

Proyecto de Fin de Máster

Integración de datos institucionales heterogéneos en plataforma abierta y distribuida

Curso 2023-2024

Este documento está realizado por:

Tonny Socorro del Pozo -

Tutor:

Benjamin López Pérez

Oviedo, a 09 de julio de 2024

Resumen

En la era de la digitalización de la información, las organizaciones se enfrentan al desafío de integrar de manera conjunta datos dispersos generados diariamente por diversas fuentes. La **Universidad de Oviedo** no es una excepción, la dispersión de los datos, provenientes de múltiples áreas dificulta obtener una visión completa y coherente de la información de la que dispone.

En este trabajo, se propone un marco de integración para establecer una plataforma unificada de almacenamiento, procesamiento y acceso a datos. Esta plataforma puede servir como un componente reutilizable para múltiples fuentes de entrada. Se valida la funcionalidad del marco con una prueba de concepto que incluye la integración efectiva de un conjunto de datos. Este trabajo se enmarca como un piloto del proyecto **Smart University(SMARTUNI),** proyecto interuniversitario en modalidad colaborativa que tiene como objetivo principal la creación de una plataforma que integre mecanismos de adquisición e ingesta de datos.

Nuestro proyecto abarca desde la planificación del sistema y el diseño de la arquitectura de información, hasta la implantación operativa de la plataforma. Se detalla la planificación del sistema de información, la descripción de la arquitectura tecnológica, la planificación y gestión del TFM, el análisis, el diseño, la construcción del sistema, las conclusiones y las posibles ampliaciones. Además, se presentan resultados de las pruebas realizadas, comprobando la efectividad del desarrollo propuesto. El sistema busca integrar datos dispersos y heterogéneos para ofrecer un medio que permita tener una visión completa y coherente que mejoren la toma de decisiones y la eficiencia institucional de la Universidad de Oviedo.

Palabras clave

Integración, ingesta de datos, interoperabilidad, orígenes de datos, recolección, transformación

Abstract

In the era of digital information, organizations face the challenge of integrating dispersed data generated daily from various sources. The University of Oviedo is no exception; the dispersion of data from multiple areas makes it difficult to obtain a complete and coherent view of the available information.

This work proposes an integration framework to establish a unified platform for data storage, processing, and access. This platform can serve as a reusable component for multiple input sources. The functionality of the framework is validated with a proof of concept that includes the effective integration of a dataset. This work is part of a pilot project for the Smart University (SMARTUNI), an inter-university collaborative project with the primary goal of creating a platform that integrates data acquisition and ingestion mechanisms.

Our project encompasses everything from system planning and information architecture design to the operational implementation of the platform. It details the planning of the information system, the description of the technological architecture, the planning and management of the TFM, the analysis, design, construction of the system, conclusions, and possible extensions. Additionally, results from the tests performed are presented, verifying the effectiveness of the proposed development. The system aims to integrate dispersed and heterogeneous data to provide a means for achieving a complete and coherent view that enhances decision-making and institutional efficiency at the University of Oviedo.

Keywords

Integration, data ingestion, interoperability, data sources, collection, transformation

Índice

Índice de tablas

Índice de ilustraciones

Capítulo 1 INTRODUCCIÓN

En la era de la digitalización de información, las organizaciones se enfrentan al desafío creciente de integrar de manera conjunta datos dispersos generados diariamente por diferentes fuentes. La **Universidad de Oviedo** no es una excepción. La dispersión de los datos, provenientes de múltiples áreas como la administración, la académica, la investigación y la infraestructura, crea una compleja **situación problemática** para su integración y consolidación. Esto dificulta obtener una visión global precisa que sea fundamental para implementar mecanismos de toma de decisiones informada y estratégica.

Para que la información generada constantemente sea útil el mundo se está inclinando rápidamente hacia el análisis de datos, lo que resalta la importancia de **la integración de datos** como un paso crucial en la canalización de procesamiento de datos. Para que se entienda, antes de un proceso de integración de datos, los datos se encuentran en distintos países y cada uno habla un idioma distinto. La integración de datos se ocupa de reunirlos en un mismo lugar y de enseñarles un mismo idioma para que puedan comunicarse.

La necesidad de unificar información provenientes de diversas fuentes requiere un proceso eficiente de recolección, transformación y almacenamiento. Sin embargo, este proceso enfrenta varios problemas significativos. Los principales desafíos incluyen la diversidad de formatos en las fuentes de datos, la rápida evolución de aplicaciones y orígenes de datos, la captura y detección de información que consume mucho tiempo, la necesidad de validar los datos ingeridos y la compresión y transformación de datos antes de su ingestión (1).

Para abordar estos problemas, es esencial llevar los datos a un formato uniforme que facilite su almacenamiento y consulta posterior. Este proceso de estandarización garantiza la coherencia y consistencia de los datos, haciéndolos fácilmente comprensibles y accesibles para todos los interesados. Lograr una ingesta y gestión de datos precisa y continua es una tarea compleja que requiere planificación adecuada, herramientas especializadas y experiencia. La investigación en ingestión de datos ha sido objeto de diversas iniciativas, como la Extracción, Transformación y Carga (ETL), la integración de datos, la deduplicación, el mantenimiento de restricciones de integridad y la carga de datos masivos (2).

Además, se subraya la importancia de la visualización de datos como un aspecto clave, ya que permite comprender tendencias, patrones y relaciones que de otro modo pasarían desapercibidos. Por lo tanto, se debe asegurar que los datos se integren y estén disponibles y accesibles para poder ser consultados y visualizados de manera eficiente y efectiva.

El **Objetivo** de este trabajo es diseñar un marco de integración, estableciendo así una plataforma unificada de almacenamiento, procesamiento y acceso de datos heterogéneos. Para cumplir con este objetivo se realiza un estudio de trabajos existentes y un análisis de los requisitos necesarios para satisfacer las necesidades, se realiza el diseño de la arquitectura y la selección de las tecnologías, se diseñan casos de uso para poner a prueba el marco, se documenta el resultado de las pruebas realizadas y se arriba a conclusiones.

Este trabajo tiene una particular relevancia práctica ya que se enmarca como un piloto del proyecto Smart University(SMARTUNI), proyecto interuniversitario en modalidad colaborativa que tiene como objetivo principal la creación de una plataforma que integra mecanismos de adquisición e ingesta de datos, facilitando la incorporación de proyectos de terceros a la plataforma.

Este proyecto piloto permitirá validar y ajustar las estrategias y tecnologías.

El trabajo documentado tiene la siguiente estructura:

- Capítulo 1: Introducción

Este capítulo introduce el contexto del proyecto, se presenta una visión general de la necesidad del proyecto y la problemática a resolver y cómo se resolverá.

- Capítulo 2: Planificación del sistema de información

Se aborda la planificación detallada del sistema incluyendo un análisis profundo de la necesidad del proyecto, una revisión bibliográfica de los trabajos existente y la identificación de los requisitos generales a cumplir.

- Capítulo 3: Descripción de la Arquitectura Tecnológica del proyecto

Se identifica la necesidad de la infraestructura tecnológica y se detalla la arquitectura tecnológica propuesta.

- Capítulo 4: Planificación y Gestión del TFM

Se presenta la planificación y gestión del trabajo fin de máster con la planificación (WBS) del proyecto, plan de riesgos y presupuesto en los diferentes momentos del desarrollo del proyecto.

- Capítulo 5: Análisis del sistema de información

Se realiza un análisis detallado del sistema donde se determina el alcance preciso del sistema, los requisitos y casos de uso a implementar.

- Capítulo 6: Diseño del Sistema de Información

Se describe el diseño del sistema, incluyendo la arquitectura detallada, los diagramas de despliegue, la descripción del diseño físico de datos y la especificación técnica de las pruebas a realizar.

- Capítulo 7: Construcción del Sistema de Información

Se detalla la construcción del sistema, la implantación y el despliegue de la plataforma y los resultados de las pruebas diseñadas.

- Capítulo 8: Conclusiones y Ampliaciones

Se presentan las conclusiones del proyecto y posibles ampliaciones futuras.

- Anexos:

  En este sección se encuentra adjunta toda la información que por cuestión de tamaño y claridad en el documento no se muestra en sus capítulos correspondientes.

Capítulo 2 PLANIFICACIÓN DE SISTEMA DE INFORMACIÓN

2.1 INICIO DEL PLAN DE SISTEMAS DE INFORMACIÓN

2.1.1 Análisis de la Necesidad

A lo largo de la historia, la información ha sido crucial para el desarrollo de cualquier proceso. Basándonos en datos cualitativos y cuantitativos, es posible representar hechos de la realidad que guían las decisiones y direcciones a seguir. Todas las organizaciones necesitan recopilar y almacenar datos de sus actividades diarias para generar conocimiento sobre el desempeño de sus procesos internos y su relación con el entorno (3).

Dicho esto, las empresas dependen cada vez más de los datos para tomar decisiones informadas, el volumen de datos generados diariamente ha crecido de manera exponencial. Este incremento ha llevado a las organizaciones a enfrentarse a grandes cantidades de datos en formatos dispares y almacenados en distintos sistemas, lo que genera tareas complejas para obtener una visión global precisa de la información almacenada, impidiendo su posterior análisis y la toma de decisiones precisas.

Para usar la información generada y almacenada de forma eficiente y global, es necesaria una integración de los datos, un proceso que, trae consigo desafíos complejos. Estos desafíos se manifiestan principalmente por la dispersión de los datos en plataformas independientes con diversas fuentes y formatos. Este proceso inicial de integración se centra en identificar estos desafíos existentes dentro de la Universidad de Oviedo.

En el contexto universitario, la complejidad de este proceso se magnifica debido a la diversidad de áreas donde se gestionan y almacenan grandes volúmenes de datos, como la gestión administrativa, académica, investigación, infraestructuras o la actividad de los miembros de la comunidad universitaria.

Por ejemplo, los registros financieros de una universidad son críticos para la gestión económica. Sin embargo, estos datos pueden estar dispersos en diferentes sistemas, como contabilidad, nóminas y compras. Esto dificulta tener una visión completa y consolidada de la situación financiera, lo que es vital para la toma de decisiones estratégicas y la planificación económica a largo plazo.

La gestión de los contadores eléctricos en los campus universitarios se recibe a través de terceros y la universidad lo recalcula internamente lo que implica recopilar y analizar datos de consumo de energía,

considerando que toda esta información se tiene en ficheros aislados y concluye en un trabajo complejo para el encargado del análisis y puede causar errores y retrasos.

Los datos relacionados con el personal, como registros de empleados, horarios y capacitación son esenciales para la gestión del personal. Sin embargo, la falta de integración con otros sistemas, como la gestión administrativa o la gestión de infraestructuras, puede dificultar la planificación de recursos humanos y la toma de decisiones estratégicas.

Algunas de las informaciones relevantes pueden estar en plataformas bajo accesos restringidos, negando el consumo a otros interesados. Esto dificulta la colaboración y el uso compartido de información crucial para la toma de decisiones.

Con la actual e inminente irrupción del IoT(Internet Of Things), este proceso se complejiza aún más. Este crecimiento ha llevado a una explosión en la generación de datos que, sumado a todo lo anterior, ha creado desafíos aún más significativos para la integración eficiente de la información. La diversidad de dispositivos y datos en el ecosistema agrega una capa adicional de complejidad, ya que cada dispositivo puede generar datos de manera diferente. Por ejemplo, algunos datos se pueden tener en documentos Word, PDF o hasta en papel, algunos dispositivos pueden generar datos en formatos estructurados, dando lugar a bases de datos, mientras que otros producen datos semiestructurados, como JSON (JavaScript Object Notation), XML (Extensible Markup Language) y CSV (Comma-Separated Values), o datos no estructurados, como texto o imágenes y otros pueden generar información en formato de código que sea necesario procesar para su comprensión. Además, los dispositivos pueden utilizar diferentes protocolos de comunicación y lenguajes de programación.

La dispersión de datos repercute directamente en la capacidad de la institución para aprovechar plenamente sus recursos disponibles y dificulta la toma de decisiones informadas y estratégicas. Sin una visión unificada de los datos, la institución se enfrenta a obstáculos en la identificación de patrones, tendencias y oportunidades clave que podrían potenciar su desempeño y eficiencia, se limita de la explotación al máximo de la información generada.

Es importante destacar que la falta de integración de datos también puede redundar en la duplicación de esfuerzos y recursos, así como en la generación de informes inconsistentes y poco confiables. Esto no solo impacta negativamente en la productividad y la eficacia operativa, sino que también puede afectar la reputación y la competitividad de la institución.

La necesidad de reconocer que una integración y por consecuencia un mejor manejo de este recurso intangible resulta en mayores beneficios para la organización. Esto lleva a establecer como objetivo estratégico la administración y provisión de información de calidad en todos los niveles de la organización.

Por lo tanto, es fundamental abordar estos desafíos mediante la implementación de un proceso de integración de datos efectivo y eficiente. Este proceso implica la unificación de datos dispersos y heterogéneos proveniente de diferentes tecnologías, plataformas y repositorios de almacenamiento; también la adopción de estándares y protocolos que garanticen la compatibilidad, el posterior almacenamiento y análisis.

2.1.2 Objetivo Principal

El objetivo principal del proyecto es definir e implementar un marco de integración sistemático que aborde la diversidad de distintas fuentes de información institucional de la Universidad de Oviedo, estableciendo así una plataforma unificada de almacenamiento, procesamiento y acceso de datos.

2.1.3 Identificación del Alcance

El proyecto se centra en el objetivo fundamental previamente definido, que implica identificar y establecer los requisitos necesarios para alcanzarlo. Como primer paso se realiza una selección de tecnologías y posteriormente se define un marco de integración que abarcará todo el proceso, desde la recepción inicial de información hasta su transformación, almacenamiento y visualización. Cada una de estas etapas es crucial. Se realiza un estudio y selección de las tecnologías más adecuadas. Se realiza un prueba de concepto con el fin de evaluar el marco de integración en desarrollo.

2.2 ESTUDIO DE INFORMACIÓN RELEVANTE

2.2.1 Trabajos relacionados

En esta sección, se revisan los avances y enfoques existentes en integración de datos, explorando los desafíos y las soluciones propuestas para garantizar un acceso uniforme a diversas fuentes de información heterogéneas y proporcionar a los usuarios una visión unificada y coherente de los datos. Por ejemplo, en (4) proponen un una solución innovadora mediante la integración virtual de datos de fuentes heterogéneas. Este enfoque, basado en una arquitectura de Mediador/Envoltura y en las especificaciones estándar de OGC SWE, ofrece flexibilidad para incorporar nuevas fuentes de datos y permite el acceso directo a las mismas en cada consulta, evitando la necesidad de infraestructura adicional de almacenamiento. El marco actualmente se está validando con datos meteorológicos y oceanográficos de dos agencias públicas en la región española de Galicia.

Los resultados muestran que el marco ofrece ventajas significativas, incluyendo un software de cliente más simple, una inversión reducida en infraestructura de gestión de datos y una integración eficiente de datos vectoriales. Además, demuestra flexibilidad en la incorporación de nuevas fuentes de datos y aprovecha las arquitecturas de hardware multinúcleo disponibles actualmente.

En el trabajo se describen los siguientes enfoques para la integración de datos:

1. Integración de datos del lado del cliente: Cada fuente de datos se accede a través de un SOS adaptado a sus propias características. Sin embargo, este enfoque puede ser complejo y requerir mucha experiencia por parte de los usuarios finales.
2. Almacén de datos: Todos los datos se integran de forma lógica en un Sistema de Gestión de Bases de Datos central con un modelo de datos común. Los procesos de ETL obtienen datos de cada fuente de datos para alimentar el almacén de datos. Esta es una buena solución para realizar análisis eficientes de datos, pero requiere infraestructura de almacenamiento y gestión adicional.
3. Integración virtual de datos: En este enfoque, el repositorio común con el modelo de datos común es virtual, y las fuentes de datos se acceden directamente en cada consulta, evitando la necesidad de nueva infraestructura de almacenamiento de datos.

La experiencia de la Universidad del Bío-Bío en (3) con su Sistema Integrado de Información Universitaria (SIUU) destaca como un caso ejemplar. Este sistema, implementado por la necesidad de modernizar los sistemas de información institucionales y superar las limitaciones de las soluciones anteriores, permitió a la universidad centralizar y consolidar datos de diversas áreas operativas. Al eliminar la duplicidad de información y facilitar el acceso en línea a datos históricos y en tiempo real, el SIUU ha mejorado significativamente la gestión administrativa, el análisis estratégico y la toma de decisiones. Este sistema ha permitido una gestión más dinámica y eficiente de procesos críticos como la administración presupuestaria, la evaluación docente y el control curricular, estableciendo un modelo de integración de datos que puede ser replicado en otras instituciones educativas para mejorar la calidad y efectividad de la gestión institucional.

En (5) se investiga el impacto de IoT en la industria eléctrica, destacando la importancia de protocolos abiertos como HTTP REST(Hypertext Transfer Protocol Representational State Transfer), MQTT(Message Queuing Telemetry Transport), LoRaWAN(Long Range Wide Area Network) y OPC UA (Open Platform Communications Unified Architecture) para lograr una plataforma IoT interoperable y eficiente. Este estudio subraya la necesidad de integración horizontal y estándares de comunicación abiertos para la Industria 4.0. De manera complementaria, en la implementación propuesta por (6) describe un sistema IoT para gestión de energía inalámbrica y monitoreo meteorológico en la agricultura, mostrando cómo la red inteligente puede optimizar el uso de energía

renovable y mejorar prácticas agrícolas. Ambos estudios resaltan la importancia de integrar datos heterogéneos y la interoperabilidad para mejorar la eficiencia y gestión en diversas industrias.

En el estudio de (7) se presenta una plataforma IoT desplegada en el campus de CEI Moncloa que incluye servicios piloto como el monitoreo del flujo de personas y el monitoreo ambiental, destacando la integración de datos de diversas fuentes. Esta plataforma cubre un área de 5.5 km2 en Madrid, utilizando sensores distribuidos en las escuelas de la UPM(Universidad Politécnica de Madrid) y hardware estándar IoT, siendo un banco de pruebas clave para servicios de Smart City. Emplea tecnologías como Wi-Fi para rastrear dispositivos móviles con anonimización de datos y sensores como el Smart Citizen Kit para medir parámetros ambientales. La plataforma integra datos mediante una arquitectura de comunicaciones híbrida y utiliza protocolos como MQTT para la transmisión y gestión de grandes volúmenes de datos, ofreciendo una aplicación web responsiva para la visualización. Esta iniciativa no solo mejora servicios y aplicaciones, sino que también fomenta el aprendizaje, la investigación y la innovación en la comunidad universitaria, demostrando la importancia de la integración de datos heterogéneos en un entorno complejo.

En el estudio de (8) se desarrolla una arquitectura de campus inteligente con tres capas: detección, datos y análisis. La capa de detección integra sensores con diferentes requisitos de energía y comunicación. La capa de datos, independiente de la plataforma, almacena valores de sensores no estructurados como pares tipo-valor, permitiendo la incorporación de nuevos datos sin cambiar el formato. La capa de análisis procesa y visualiza los datos utilizando herramientas abiertas y propietarias. Los despliegues piloto demostraron beneficios en la toma de decisiones automatizada y continua, destacando la heterogeneidad de fuentes y formatos de datos como un desafío clave. Se implementaron soluciones como contadores de haces, cámaras de detección de presencia, tecnología IoT para medir factores ambientales, sistemas de gestión de estacionamiento y sensores ultrasónicos con LoRaWAN para medir colas de pasajeros en paradas de autobús, complementados con registros de conexiones Wifi. Destacan los beneficios que la recopilación de datos continua y automatizada aporta a la toma de decisiones en un campus universitario, y se pueden traducir en entornos más grandes similares a ciudades inteligentes

En (9) se estudia el caso de SmartUA, donde la Universidad de Alicante adoptó un enfoque hacia la universidad inteligente en 2014. Implementaron una metodología basada en Arquitectura Orientada a Servicios para desacoplar y reacoplar componentes de TI (Tecnología de la información). Se diseñó una arquitectura específica con capas y subcapas, incluyendo monitoreo, negocio, presentación, servicio y comunicaciones. La capa de Monitoreo recopila datos de sensores, cámaras, RFID (Radio Frequency Identification) y dispositivos móviles, procesados en la capa de Negocio para generar servicios inteligentes. La capa de Presentación muestra la información de manera comprensible,

mientras que la capa de Servicio proporciona funciones comunes y la capa de Comunicaciones facilita la interacción entre componentes. Se desarrollaron servicios como Wi-Fi, consumo de electricidad y recarga de vehículos, apoyando la investigación académica con una API (Application Programming Interface) segura y anónima.

El estudio (10) presenta una optimización del sistema de integración de datos de la plataforma digital de un campus, basada en XML, para abordar la ineficacia en el procesamiento de datos heterogéneos locales. Se mejoran diversos aspectos del sistema, desde la configuración del hardware hasta el proceso de conversión e integración de datos. Se emplean algoritmos de programación y se normalizan los datos heterogéneos para lograr una integración eficiente. Los resultados experimentales muestran un aumento significativo en la tasa de integración de datos, superando en más del 20% al sistema tradicional. Esta investigación, presentada en la Conferencia Internacional de 2019 sobre Transporte Inteligente, Big Data y Ciudad Inteligente, destaca la importancia de la tecnología XML en la gestión de datos del campus digital y su potencial para mejorar la eficiencia en el procesamiento de datos heterogéneos.

El estudio (11) propone un método inteligente de integración de datos para la regulación del tiempo compartido de la carga eléctrica elástica, abordando la ineficiencia y la redundancia en la integración de grandes datos. Se calcula el estado paralelo de los datos y se analiza su descripción matemática, optimizando el proceso de integración para mejorar la precisión y eficiencia. Los resultados experimentales demuestran una integración más precisa, una reducción de datos redundantes y una mejora en la eficiencia. Este enfoque se destaca por su capacidad para reducir la redundancia de datos, mejorar la precisión y reparar datos inconsistentes, mostrando un gran potencial para aplicaciones futuras en el análisis y la toma de decisiones en el despacho de energía eléctrica.

El reciente estudio (12) ofrece una visión integral sobre la integración de datos en proyectos de análisis de datos. Destaca la importancia de la integración de datos en la recopilación y análisis de datos de diversas fuentes heterogéneas, como API web, bases de datos relacionales y no relacionales, entre otras. El estudio compara los enfoques tradicionales de ETL con el enfoque emergente de ELT y analiza cómo ELT con MPP (Procesamiento Masivamente Paralelo) puede mejorar la eficiencia y el rendimiento del análisis de datos. Además, explora la migración de datos de bases de datos heredadas a la nube y examina las diferencias críticas entre ETL y ELT en términos de procesamiento y rendimiento. Finalmente, presenta un caso de estudio en la industria del transporte marítimo para ilustrar la implementación práctica de un trabajo de ELT para cargar datos en tablas de almacén de datos. Este estudio proporciona valiosas perspectivas sobre cómo las herramientas ETL y ELT pueden facilitar la integración de datos para aplicaciones exitosas de ciencia de datos.

2.3 IDENTIFICACIÓN DE REQUISITOS DEL SISTEMA

Esta sección identifica los requisitos fundamentales para la plataforma a implantar.

2.3.1 Análisis de requisitos

Para lograr una integración de datos efectiva, es esencial enfrentarse a una serie de retos que hacen de este un desafío complejo.

El marco de integración debe ser lo suficientemente versátil como para funcionar con una variedad de fuentes de datos y aplicaciones de alto nivel. Esto implica la capacidad de obtener flujos de datos desde una diversa variedad de orígenes, como HTTP/Web Sockets API, REST API, Streaming API, IoT Hubs o colas de mensajes incluyendo formatos convencionales cómo Excel, PDF, Word, JSON, XML, CSV o cualquier tipo de documentación.

La tarea más compleja en este escenario es diseñar e implementar los flujos de procesamiento de los datos para lograr un almacenamiento estandarizado, lo que facilita su posterior análisis. Esta transformación es esencial para asegurar la coherencia y la integridad de los datos, así como para optimizar su almacenamiento y manipulación en el sistema. Por lo tanto, es fundamental contar con la capacidad de realizar transformaciones sobre los datos para satisfacer diversos tipos de procesamientos, p.ej., limpieza de datos, normalización, enriquecimiento, agregación, división.

Las múltiples fuentes con tasas de llegada de datos variables imponen una demanda cambiante de recursos en el sistema. Por lo tanto, es imperativo que el marco sea escalable para poder manejar volúmenes crecientes de datos. En casos como este cuando los datos llegan muy rápido, los sistemas a veces se sobrecargan. Por eso, es crucial que nuestro sistema pueda almacenar esos datos extras en momentos de mucha actividad, y también reproducirlos más tarde cuando sea necesario.

Dada la importancia de los datos y la posibilidad de operar en un entorno distribuido, donde factores como la latencia de la red, la concurrencia de accesos y las fallas de los nodos pueden afectar la estabilidad del sistema, es esencial que el sistema incorpore mecanismos de alta disponibilidad. Esto asegura la integridad de los datos ante incidencias en el sistema.

Las capacidades efectivas de procesamiento de mensajes son esenciales para garantizar el flujo continuo de datos a través de la infraestructura de transmisión, lo que asegura un flujo de trabajo sin interrupciones y una ingesta eficiente de datos en los sistemas de destino.

2.3.2 Definición de requisitos

Los requisitos identificados para satisfacer las necesidades de la plataforma a implantar son los siguientes:

Escalabilidad: Capacidad para aumentar la capacidad de procesamiento y almacenamiento según las demandas del sistema, permitiendo manejar volúmenes masivos de datos de manera eficiente y escalable.

Captura de Diversas Formatos de Datos: Capacidad para capturar una amplia gama de fuentes de datos, incluyendo diversos formatos y estructuras.

Integración con Servicios Externos: Facilidad para conectarse con servicios externos para enriquecer y ampliar las capacidades del sistema. Esto incluye la comunicación fluida y el acceso a datos desde y hacia múltiples fuentes y destinos

Transformación de Datos: Capacidad y flexibilidad para realizar diversas transformaciones en datos entre múltiples formatos y tipos.

Procesamiento de Datos en Tiempo Real: Habilidad para analizar y procesar datos en tiempo real para obtener información instantánea y acciones rápidas.

Disponibilidad: Garantizar que el sistema esté disponible y operativo de manera continua, garantizando la tolerancia a fallos.

Seguridad: Garantizar la protección de los datos durante el proceso.

Capítulo 3 DESCRIPCIÓN DE LA ARQUITECTURA TECNOLÓGICA DEL PROYECTO

En este capítulo, se aborda la descripción de la arquitectura tecnológica. Se define la estructura y disposición de los componentes tecnológicos que se utilizan en la implantación del proyecto.

3.1 IDENTIFICACIÓN DE LAS NECESIDADES DE INFRAESTRUCTURA TECNOLÓGICA

En esta subsección, se lleva a cabo un análisis detallado para identificar las necesidades específicas de la infraestructura tecnológica requerida para el sistema de información. Para comprender las necesidades, es fundamental analizar la definición de requisitos detallados en el capítulo 3.4 "Identificación de Requisitos del Sistema" de este documento.

Basándonos en el análisis realizado, se identifican las siguientes necesidades de infraestructura tecnológica:

1. **Open Source:** Se necesitan herramientas de código abierto.

2. **Entornos Necesarios**: Se requiere un entorno de procesamiento distribuido y escalable que pueda manejar grandes volúmenes de datos de manera eficiente. Este entorno debe ser capaz de adaptarse dinámicamente a cambios en la carga de trabajo y garantizar la disponibilidad y la integridad de los datos.
3. **Conectividad y Comunicaciones**: Es crucial contar una infraestructura de red robusta y confiable que permita la comunicación fluida entre los diferentes componentes del sistema, incluyendo la captura y procesamiento de datos y la integración con servicios externos. Se deben considerar protocolos de comunicación eficientes y mecanismos de redundancia para garantizar la disponibilidad y la tolerancia a fallos.
4. **Disponibilidad**: La infraestructura tecnológica debe poder garantizar altos niveles de disponibilidad y continuidad operativa. Esto implica recuperación ante fallos para garantizar la integridad de los datos en todo momento.
5. **Servicios Críticos**: Se deben contar con los servicios indispensables necesarios para el funcionamiento del sistema, como la seguridad, la autenticación, la autorización y el monitoreo. Estos servicios deben estar integrados de manera transparente en la infraestructura tecnológica para garantizar la protección y el control adecuados sobre los datos y los recursos del sistema.
6. **Flexibilidad**: La infraestructura debe ser capaz de manejar múltiples formatos, protocolos y orígenes de manera flexible y ágil.
7. **Datos concurrentes**: La infraestructura debe ser capaz de soportar la concurrencia de usuarios trabajando con los mismos datos de manera efectiva.
8. **Procesamiento de datos**: la infraestructura debe ser capaz de realizar diversa operaciones de procesamiento de la información por ejemplo agregación, modificación, eliminación, división o enriquecimiento de los datos.
9. **Exploración de datos:** Se necesita la capacidad de extraer, filtrar y buscar información para la creación de visualizaciones que faciliten la explotación de la información generada.
10. **Despliegue:** Todos los componentes de la solución deben poder desplegarse en contenedores con el fin de garantizar la portabilidad y escalabilidad.

3.1.1 Estudio de alternativas tecnológicas

En esta sección se abordará la evaluación y comparación de diversas herramientas y tecnologías relevantes para el proyecto en los diferentes procesos. Se analizarán detalladamente las características, capacidades y limitaciones de cada herramienta, centrándose en su capacidad para satisfacer los

requisitos específicos de infraestructura tecnológica identificados previamente. Además, se revisarán casos de uso, estudios de rendimiento y experiencias previas de implementación. Este análisis exhaustivo permitirá identificar las alternativas más adecuadas que cumplan con los objetivos del proyecto.

3.1.1.1 Herramientas de recolección y procesamiento de datos

En el trabajo de (13) sobre la monitorización eficiente de redes ópticas en entornos 5G/6G, se utilizó Apache NiFi como herramienta principal para automatizar el procesamiento y distribución de grandes volúmenes de datos de transceptores ópticos. La elección de Apache NiFi se basó en su capacidad para absorber flujos masivos de datos de dispositivos fotónicos y redes ópticas de alto rendimiento y alta densidad, optimizando así el rendimiento del sistema. Los resultados obtenidos demostraron una mejora significativa en el rendimiento del procesamiento de datos, tanto en términos de tiempos de consulta para una sola entrada como en tiempos de procesamiento, lo que resulta crítico para identificar eficazmente problemas en las redes ópticas. Además, se observó que el tiempo de procesamiento de datos disminuyó de manera no lineal a medida que aumentaba el número de nodos en el clúster, lo que sugiere un mayor potencial de escalabilidad del sistema.

En (14) utilizaron varias herramientas en el contexto de la automatización y ejecución de modelos científicos en un entorno de simulación coordinada. Entre estas herramientas se incluyen Docker para la virtualización de contenedores, Apache NiFi para la ingestión y transformación de datos, y Apache Kafka como canal de mensajes. Estas herramientas fueron elegidas para abordar la necesidad de ejecutar modelos independientes como procesos separados en contenedores Docker, permitiendo la ejecución de flujos de trabajo de simulación coordinada en un clúster informático escalable. Resaltan que al integrar Nifi el marco diseñado permite a los usuarios crear, ejecutar y gestionar fácilmente flujos de trabajo de procesamiento de datos o simulación a través de la interfaz de usuario web de NiFi. Los resultados obtenidos mostraron una mejora significativa en la automatización y ejecución de flujos de trabajo de procesamiento y simulación de datos científicos, reduciendo la carga para los modeladores y expertos en dominios no relacionados con la informática y facilitando la comunicación entre los diferentes módulos y servicios asociados.

En (15) se realiza un estudio sobre herramientas de Big Data Ilustración 1, se analizan diversas herramientas clave en cada fase del proceso, como adquisición, almacenamiento, procesamiento y análisis de datos. Entre las herramientas evaluadas se encuentran Apache NiFi, Apache Flume, Apache Kafka, Apache Cassandra, Apache Spark, Redis y Apache Flink. Para investigar la interoperabilidad y el rendimiento, se llevan a cabo experimentos comparativos. Se realizó una comparación del uso de

recursos entre las herramientas de ingestión de datos NiFi y Flume. Los resultados de los experimentos comparativos que el rendimiento de las dos herramientas es similar y que Flume tiene un uso de memoria considerablemente menor que NiFi. Sin embargo, NiFi tiende a generar intercambios de memoria y bloqueos cuando la memoria está restringida a 3 Gb. En cuanto al uso de la CPU, es similar para ambas herramientas, aunque el uso de la CPU de NiFi disminuye cuando la memoria es limitada. En cuanto a la duración del linaje, los registros se procesan en milisegundos en Flume, mientras que en NiFi puede llevar hasta 20 segundos debido a la contrapresión aplicada por NiFi, lo que puede aumentar el uso de memoria. Estos resultados sugieren que cada herramienta tiene diferentes requerimientos de recursos y funcionan de manera distinta para una misma configuración. Flume parece ser más eficiente para la tarea en cuestión, aunque el mayor uso de recursos de NiFi se justifica por sus diferencias conceptuales. NiFi ofrece una interfaz web para facilitar la manipulación y el monitoreo de datos, mientras que Flume tiene una arquitectura más ligera y se configura mediante archivos de configuración. El experimento concluye en que ambas herramientas se adaptan mejor a diferentes escenarios, y los diseñadores deben considerar el uso de recursos y los requisitos conceptuales al seleccionar la herramienta más adecuada para sus necesidades.

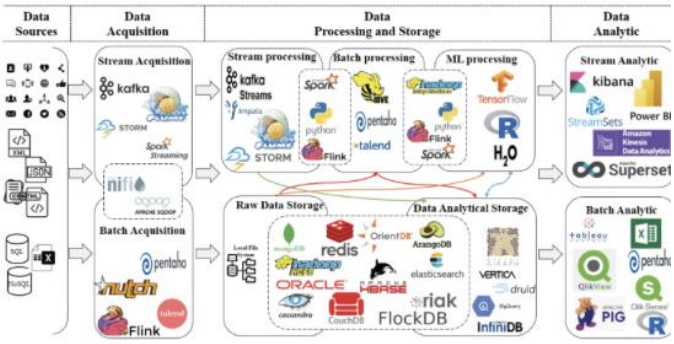


Ilustración 1. Herramientas populares por proceso(15)

En (16) detalla el proceso de ingestión de datos en el contexto de big data y se analizan herramientas de ingestión y preparación de datos. Se discute la selección de herramientas para ayudar a los usuarios a elegir la más adecuada, y se revisa la comparación de herramientas de preparación de datos. Para completar este proceso eficazmente, se consideran varios parámetros y se muestran en Ilustración 2
En un caso de uso específico de Flume, se utilizó esta herramienta para detectar variaciones en los datos de calefacción doméstica en el sistema de calefacción a vapor municipal de Jinan, donde los sensores instalados en todas las habitaciones recopilaron información sobre la potencia térmica, el calor acumulado y la temperatura. Los datos de los sensores fueron procesados con Spark para obtener resultados específicos.

Por otro lado, Kafka se destaca como una herramienta de transmisión distribuida que ofrece un sistema unificado de alimentación de datos en tiempo real y mensajería, con baja latencia y tolerancia a fallos. Por último, NiFi se describe como una herramienta que proporciona una interfaz de usuario web amigable para construir flujos de datos visualmente en tiempo real, con capacidad para manejar grandes volúmenes de datos, priorizar datos y rastrear su procedencia, centrándose en la seguridad mediante el uso de protocolos seguros para garantizar la integridad de los datos.

Criteria	Sqoop	Flume	NIFI	Kafka
latest stable release	1.4.7	1.9.0	1.10.0	2.4.0
Primary written in	Java	Java	Java	Java
License	Open Source	Open Source	Open Source	Open Source
Basic nature	works well with any RDBMS that has JDBC(Java Database connectivity) like oracle	works well for streaming data sources which continuously generating.	works well for data flow creation between different systems.	works well for messaging Streaming data
Type of Data	Batch Data	stream Data	Batch and Stream Data	Stream
Type of loading	Not-event driven	Event driven	Both (Event and not-event)	Event driven
Architecture	Connector based	Agent based	Flow based	Process topology .
Link to HDFS	Connected	Connected	Connected	Programmable
Direction based on HDFS	Bi-directional	Uni-Directional (into Hadoop)	Bi-directional	-
User Interface	Shell command line	Shell command line	Web user interface	Shell command line
Data compression	Supported	Supported	Supported	Supported
Event prioritization	-	Supported by (Failover sink processor) concept	Supported	Programmable
Back pressure	-	No	Yes	Yes
Notable users	1-Apollo Group 2-Coupons.com	1-Meebo . 2-Sharethrough . 3-SimpleGeo.	1-Macquarie Telecom 2-Group. 3-Hastings Group. 4-Payoff.	1-Uber 2-Booking.com 3-Slack.

Ilustración 2. Comparación de parámetros(15)

En el estudio de (1) analizan las características y el rendimiento de algunos de las herramientas de ingesta de Big Data ampliamente utilizadas. El análisis se realiza para tres herramientas de ingesta de datos, desarrollado por Apache: Flume, Kafka y NiFi. El estudio se basa en la experiencia de personas que han utilizado las herramientas. Los parámetros usados para la comparación fueron velocidad para procesar datos de diferentes fuentes, el tamaño, la frecuencia y el formato de los datos. El análisis implica dos acciones principales: la ingestión de datos y el procesamiento a lo largo del cual se verifica qué tan rápido puede consumir el sistema datos de diversas fuentes y procesarlos. Se destacan mediciones de velocidad, número de archivos procesados por segundo, escalabilidad y durabilidad del mensaje como indicadores clave. Para Flume, se presenta un estudio de rendimiento que demuestra su capacidad para manejar aproximadamente 70,000 eventos por segundo en un entorno de un solo equipo físico, sin pérdida de datos. En términos de funcionalidad, se compararon las herramientas desde la perspectiva de su capacidad para recolectar y mover grandes cantidades de datos de registro de manera eficiente y confiable. Se concluye que, en términos de rendimiento, Kafka ofrece los mejores resultados, mientras que en términos de funcionalidad, NiFi es la mejor opción.

En un estudio reciente realizado por (17) se utilizó la herramienta Apache NiFi para la ingestión de archivos con el fin de comprender su rendimiento en diferentes configuraciones. Se comparó y verificó el rendimiento de NiFi para la ingestión tanto en sistemas locales como en la nube, utilizando diferentes procesadores. La ingestión en sistemas locales se llevó a cabo en dos máquinas diferentes, mientras que la ingestión en la nube se validó en plataformas principales como Google Cloud y AWS Cloud. Los resultados y gráficos obtenidos indicaron que el rendimiento de NiFi para la ingestión en la nube fue comparable, demostrando ser una opción robusta para los requisitos de ingestión en la nube.

En (18) presentan una canalización de datos que orquesta Apache NiFi (NiFi), Apache MiNiFi (MiNiFi) y otras herramientas como una solución automatizada para transmitir y archivar los datos de la información espacial como la detección y el rango de la luz (LiDAR) capturados por los dispositivos perimetral desplegados en Nevada. En el diseño de la canalización de datos consideran la automatización del envío de datos LiDAR en archivos comprimidos con un esquema de nombre establecido. La canalización se diseñó para ser escalable y conservar recursos como ancho de banda, CPU y almacenamiento. Se ejecuta en tres hosts diferentes: los ordenadores perimetrales en el lado de la calle, el centro de datos de UNR y el clúster HPC de UNR Pronghorn. Para la a implementación de la canalización, utilizan Docker Compose en el borde y Kubernetes en el centro de datos. Emplean NiFi, MiNiFi y tcpdump en contenedores Docker para gestionar la transferencia y compresión de los datos. Los resultados de muestran que NiFi y MiNiFi son eficientes en la transferencia de datos LiDAR. También presentan una comparación entre las características de Globus, RSync y el enfoque utilizando NiFi y MiNiFi. Globus utiliza GridFTP para transferir datos y es ideal para archivos grandes, con capacidad para secuencias de red paralelas que permiten una transferencia más rápida en comparación con RSync y nuestro método. Mientras que los tres enfoques admiten la verificación de archivos, NiFi y MiNiFi necesitarían una implementación específica para esta función dentro de su flujo de datos. Globus facilita la expansión de puntos finales y utiliza software en la nube para manejar la transferencia de datos entre diferentes fuentes. NiFi es escalable mediante el agrupamiento con Zookeeper y permite enviar datos a múltiples tipos de destinos, lo que brinda flexibilidad para realizar análisis o enviar datos a otras fuentes como almacenes en frío. Aunque NiFi puede requerir tiempo de configuración y no ofrece la misma facilidad de uso de la interfaz de usuario que Globus para la transferencia de archivos, su capacidad para personalizar flujos de datos y su flexibilidad lo hacen destacar en comparación con RSync y Globus. Concluyen que el enfoque utilizando NiFi y MiNiFi es prometedor, con posibilidades de mejorar la calidad y el análisis los datos.

En (19) se presenta un flujo de datos automatizado entre sistemas utilizando varias herramientas entre las que se encuentra: Kafka y ZooKeeper para alta disponibilidad y tolerancia a fallos; Apache NiFi para procesar los datos, dividir archivos JSON y redirigirlos según su contenido; los resultados se

envían a Apache Spark para procesamiento estructurado en streaming, y finalmente se almacenan en una base de datos MongoDB. Destacan las capacidades de detección de lagunas en los datos en tiempo real, algo crucial para la precisión de los análisis, especialmente considerando la frecuencia incremental de generación de datos y se enfatiza la escalabilidad y fiabilidad del sistema. Se discute la importancia del procesamiento de datos en streaming frente al enfoque de procesamiento por lotes. Se resalta cómo estas tecnologías abordan estos desafíos, permitiendo la recolección, procesamiento y análisis efectivos de grandes volúmenes de datos en tiempo real.

En el trabajo de (20) presenta un sistema prototipo para experimentos con el procesamiento de flujos de datos en tiempo real de dispositivos IoT utilizando Apache NiFi, Apache Kafka, Apache Storm, Node-RED y Swagger. El sistema integra estas tecnologías para gestionar la adquisición, procesamiento, almacenamiento y visualización de datos IoT. Las herramientas se despliegan en un entorno Ubuntu 18.04. Los resultados experimentales muestran cómo se diseña el flujo de trabajo de datos con Apache NiFi, la implementación del procesamiento de datos con Apache Storm, y la creación de la recopilación de datos y el panel de control con Node-RED. Se describe cómo se almacenan los datos en MongoDB y se accede a ellos a través de una API diseñada con Swagger.

En el trabajo de (21) presenta un estudio comparativo detallado de la migración de datos utilizando Apache Spark y otras herramientas y marcos populares como NiFi, Talend, Kafka, Hadoop y A WS Glue. Los resultados de la investigación destacan que Apache Spark sobresale en términos de tiempos de migración de datos más rápidos, superando significativamente a las otras herramientas. Además de la velocidad de migración, también evaluaron la tasa de transferencia de datos de cada herramienta. Aquí, Apache Spark registró la tasa más alta, superando a las demás herramientas. En términos de tolerancia a fallos y capacidad de recuperación, Apache Spark nuevamente mostró fortaleza al recuperarse rápidamente de fallos y continuar con la migración de datos sin interrupción. NiFi también demostró buenas capacidades en este aspecto, aunque con tiempos de recuperación ligeramente más lentos que Spark. Otro aspecto importante evaluado fue la facilidad de uso y flexibilidad. Mientras que Apache Spark ofrecía una alta flexibilidad y personalización, su curva de aprendizaje era más pronunciada. En cuanto a escalabilidad, Apache Spark y Hadoop demostraron ser altamente escalables, mientras que otras herramientas mostraron dificultades con volúmenes de datos más grandes. Además, Apache Spark y NiFi destacaron por sus capacidades integradas para manejar fallos y garantizar la integridad de los datos. En conclusión, aunque Apache Spark se destacó como la mejor opción en la evaluación, aunque se identificaron situaciones específicas donde otras herramientas pueden ser más adecuadas, como NiFi para casos de uso que requieran transmisión en tiempo real o integración de datos.

En (22) proponen un marco de ingestión de datos de transmisión ligero, de alto rendimiento y escalable para el procesamiento de datos de transmisión donde usan utilizan una combinación de Streamsets Control Hub en lugar de Apache NiFi y Kafka para la distribución de flujos de datos utilizando la API de noticias globales. El estudio se enfoca en monitorear titulares de noticias de última hora y evalúa el rendimiento de ambas herramientas en términos de eficiencia de carga de datos y procesamiento. Establecen un escenario de caso de uso que incluye la ingestión de titulares de noticias de última hora, la eliminación de ruido en los flujos de datos y la distribución de las noticias a motores de análisis o al almacén de datos permanente. En la evaluación del rendimiento, se compara la eficiencia de carga de datos entre StreamSets Data Collector y Apache NiFi utilizando flujos de datos de la API de noticias. Los resultados revelan que StreamSets Data Collector supera a Apache NiFi en términos de rendimiento de carga de datos. El autor resalta que Apache NiFi y Kafka se han utilizado ampliamente para este propósito y que Apache NiFi puede presentar limitaciones en términos de rendimiento y latencia debido a la intensidad en E/S de sus procesadores.

En (2) se presenta un marco de gestión de flujo de datos escalable y tolerante a fallos, diseñado para ser un componente reutilizable en diversas fuentes de datos estructurados y no estructurados y se demuestra la utilidad del marco mediante un estudio de caso de procesamiento de flujos de datos del mundo real que integra Kafka y HDFS en un sistema impulsado por NiFi. El marco propuesto se observa en Ilustración 3. Cómo resultado de la evaluación experimental se concluye que el marco basado en NiFi, Kafka y HDFS, demostró ser una solución efectiva y escalable para la ingestión y gestión de flujos de datos a gran escala. Se identificaron las siguientes conclusiones específicas sobre las herramientas utilizadas:

NiFi: Si bien NiFi mostró ser una herramienta versátil y fácil de usar, se observó que algunos de sus procesadores pueden ser intensivos en CPU, E/S y/o memoria. Esto puede causar cuellos de botella de rendimiento en entornos de flujos de datos de alto volumen y alta velocidad. Sin embargo, se destacó su capacidad para rastrear y evaluar datos a lo largo de la ruta de flujo, así como su función de linaje de datos y contrapresión para gestionar la carga de trabajo.

Kafka: Se demostró que Kafka es una opción robusta para la distribución de flujos de datos a sistemas descendentes. Su capacidad para manejar grandes volúmenes de datos en tiempo real y su arquitectura distribuida lo hacen adecuado para entornos donde se requiere un alto rendimiento y tolerancia a fallos.

HDFS: La integración con HDFS proporcionó una solución confiable para la persistencia de datos a largo plazo. La capacidad de almacenar datos de forma distribuida y escalable es fundamental para el procesamiento de grandes volúmenes de datos en entornos de big data.

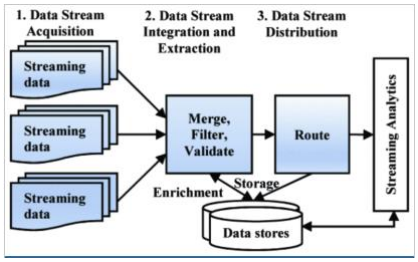


Ilustración 3. Marco propuesto por Isah & Zulkernine(2)

3.1.1.2 Herramientas de visualización

Además de las herramientas de visualización tradicionales, como Tableau y Power BI, se han desarrollado soluciones especializadas para abordar las necesidades específicas de la visualización de datos. En el trabajo de (23) se presenta la tecnología ELK (Elasticsearch, Logstash y Kibana) como una solución integral para procesar datos de IoT utilizando dispositivos instalados en el edificio de ABB Corporate Research en Cracovia como caso de estudio, aunque estas herramientas fueron originalmente diseñadas para el manejo de grandes volúmenes de datos de registro, su versatilidad les permite almacenar, buscar y visualizar una variedad de información, incluidos los datos generados por dispositivos IoT. En el trabajo se presenta una plataforma unificada para acceder, almacenar y visualizar datos institucionales. El autor discute entra varias alternativas a la pila ELK, como Splunk, Graylog y Grafana. Finalmente, el autor concluye (24) que la solución propuesta proporciona flexibilidad y funcionalidad para el análisis de datos, explica que Splunk es una solución completa y rica en características, pero no es de código abierto. Graylog es una alternativa de código abierto que utiliza Elasticsearch para el almacenamiento de datos y MongoDB como almacén de metadatos. Grafana, aunque no está específicamente diseñado para el procesamiento de registros, ofrece una interfaz web de código abierto similar a Kibana y se utiliza principalmente para monitorear métricas de rendimiento de la infraestructura de TI, también soluciones basadas en la nube, como Loggly, Sumo Logic y PaperTrails, que están diseñadas principalmente para el procesamiento de registros, pero pueden tener aplicaciones limitadas en el análisis de datos de IoT.

En el estudio de (24) se presenta un experimento utilizando ELK para realizar un análisis de sentimientos en datos de tweets recopilados a través de la API de transmisión de Twitter. Se utiliza Logstash para cargar los datos de tweets en Elasticsearch y Kibana para visualizar los resultados del análisis de sentimientos. El proceso implica la creación de una aplicación en la plataforma de desarrolladores de Twitter para obtener acceso a la API de transmisión de Twitter. Luego, se filtran los tweets por una palabra clave específica y se realiza un análisis de sentimientos. Los resultados del análisis se agregan a Elasticsearch y se visualizan en Kibana a través de gráficos circulares que

muestran el porcentaje de sentimientos positivos, negativos y neutrales de los tweets recopilados. Cómo resultado del experimento el autor concluye que ELK Stack es una herramienta eficiente y conveniente para realizar análisis de sentimientos en datos de redes sociales como Twitter. Además, se destaca su eficacia en la gestión de registros, su rentabilidad y su capacidad para funcionar bien con grandes conjuntos de datos, lo que lo hace adecuado para una variedad de aplicaciones, incluido el monitoreo de actividades ilegales y la seguridad en el Internet de las cosas.

En (25) presentan una aplicación de tecnologías avanzadas de Big Data para el análisis visual de los datos de calidad del aire. Spark, Elasticsearch Engine y Kibana están bien combinados para producir una visualización espaciotemporal interactiva a través de gráficos verticales y de líneas, mapas de coordenadas y gráficos de mapas de calor. El autor resalta que los principales avances de la solución en gran medida son el resultado de la pila tecnológica utilizada. Basada en Elasticsearch, la solución utiliza el formato JSON para la ingestión de datos. Elasticsearch también proporciona una API RESTful flexible para la comunicación con las aplicaciones de los clientes. Además, a través de Elastic Beats, un conjunto de remitentes de datos ligeros que permiten enviar datos convenientemente a Elasticsearch, se podrían ingerir diferentes tipos de formatos de datos. La solución admite diferentes gráficos de visualización que proporcionan el beneficio de tener una visión mucho más holística de los datos. Por último, pero no menos importante, la pila tecnológica de ELK proporciona características avanzadas para combinar diferentes gráficos en un panel de control integral de visualización de datos.

En (26) se utiliza ELK Stack como una solución efectiva para la gestión de registros. El autor describe la herramienta y hace énfasis en lo siguiente. La visualización es otro espacio dondequiera que sobresale. La pila elástica incluye una herramienta conocida como Kibana que hace gráficos, gráficos y diferentes visualizaciones valiosas a partir de la información. Mongo carece de una herramienta de suministro abierto de alta calidad para hacer visualizaciones, y lo mismo ocurre con MySQL. La mayoría de las soluciones que se ofrecen son prohibitivamente caras y también la pila elástica es gratuita. Es fácil usar estas herramientas para bombear información de esas tiendas de información a Elasticsearch y así producir visualizaciones con Kibana. Llevándolo a concluir que “En las herramientas de observancia de registros, se afirma que la combinación de Logstash, Elasticsearch y Kibana es la herramienta más efectiva”.

En (27) se presenta la implementación de un entorno de Centro de Operaciones de Seguridad (SOC) utilizando la herramienta SIEM (Security Information and Event Management) ELK Stack para la recopilación, procesamiento y visualización de datos de registros de Twitter. Se describe el proceso de extracción de datos de Twitter mediante la API, su procesamiento a través de ELK y su visualización en Kibana. Cómo resultado el autor concluye en que hay algunas características integradas que faltan en ELK. La versión libre de pila de ELK no termina toda la caja de herramientas que necesita el analista

de seguridad. Faltan capacidades de alerta, reglas de correlación (conjunto de reglas que las funciones activan), etc.

En (28) Se menciona que la solución de big data puede ayudar a reducir el tiempo de análisis de registros para responder a los ataques, pero los productos comerciales son costosos para las pequeñas y medianas empresas. Por lo tanto, se sugiere el uso de plataformas de código abierto, como ELK Stack, para esta tarea. Se detallan las limitaciones del producto comercial Splunk y se presenta la viabilidad de ELK Stack mediante un análisis comparativo de rendimiento en el procesamiento de análisis de registros. Se concluye que ELK Stack puede ser una poderosa herramienta de análisis de registros de seguridad inicial con un rendimiento aceptable en comparación con un producto comercial de alto costo.

En (29) se presenta la arquitectura de un prototipo para un sistema de gestión de confort, utilizando la pila de código abierto ELK. La arquitectura admite sensores y protocolos heterogéneos. Se basa en hardware y software de código abierto, lo que da la ventaja de ser de bajo costo y fácil de mantener. Se presentó un bando de pruebas para explicar la viabilidad del concepto teórico de gestión del confort, que consistía en sensores heterogéneos para la detección de parámetros en interiores y exteriores.

En (30) se presenta una propuesta para construir e implementar un sistema basado en Elastic Stack para procesar datos de diversos tipos y formatos, incluyendo eventos de aplicaciones personalizadas, con el objetivo de transformar uniformemente los datos de entrada. Para la implementación práctica del sistema en un centro de datos, se recopilan eventos y registros de varios dispositivos IoT a través de puertas de enlace LoRa WAN, puertas de enlace de mensajería y subsistemas de gestión de información. Los datos se envían directamente o a través de Beats, y se utilizan Logstash y Apache Kafka para el procesamiento y almacenamiento en búfer de flujos de datos, la ejecución de diferentes filtros y mejoras en la transformación de datos de entrada, así como para indexar y almacenar datos en Elasticsearch. Kibana proporciona capacidades de visualización complementarias a Elasticsearch. El sistema está diseñado y configurado como una solución elástica multi-inquilino para dirigirse a diferentes consumidores mediante índices separados a diferentes espacios y con controles de acceso basados en roles y atributos. Se discuten tecnologías alternativas, como Apache Solr y Splunk, y se compara la solución propuesta con el servidor Prometheus y Grafana. Se destaca que, aunque Prometheus es eficiente para almacenar métricas y cuenta con mecanismos de alerta integrados, está especializado en medir un solo tipo de datos y no está optimizado para ser un almacén de métricas a largo plazo. En contraste, la solución propuesta combina varios tipos de datos operativos y debe escalar a medida que los datos crecen, lo que hace que Prometheus no sea un reemplazo adecuado para Elasticsearch. El autor resalta los desafíos de la administración compleja y el mantenimiento de la arquitectura elástica.

En (31) se estudia una solución optimizada en tiempo real que utiliza la visualización de datos para todos los problemas de predicción asociados con el sistema de reserva de taxis en línea. En el trabajo proponen realizar la visualización con Kibana junto con Apache Flink y Elasticsearch. El resultado de este estudio es computacionalmente eficiente en términos de tiempo y memoria, mientras que también es aplicable a todos los dispositivos de mano.

En (32) proponen un nuevo sistema de procesamiento y almacenamiento de registros que llaman Loginson, que está dirigido específicamente al procesamiento, almacenamiento y visualización de cantidades masivas de registros. La novedad de Loginson radica en abordar todos los requisitos anteriores a una velocidad de ingestión de registros muy altas. En cuanto a la representación gráfica, se discuten herramientas como D3.js, Grafana y Kibana, concluyendo que Kibana es la opción preferida ya que tiene una variedad de gráficos (gráficos circulares, series temporales, histogramas, etc.) y una API de navegación de datos para los índices de Elasticsearch. Kibana y Elasticsearch están en constante desarrollo por parte del equipo de Elastic, a diferencia de Grafana, también admite la incrustación de paneles y gráficos que permiten páginas web personalizadas.

En el estudio reciente (33) se evalúa la capacidad de herramientas modernas de Business Intelligence (BI) para manejar grandes conjuntos de datos biológicos. Se analizan cinco herramientas de BI, incluyendo Kibana, Siren Investigate, Microsoft Power BI, Salesforce Tableau y Apache Superset, en cuatro estudios de caso exploratorios. Destaca la versatilidad de Kibana para conectarse con Elasticsearch, importar datos biológicos, y crear paneles interactivos para visualizar y analizar datos detallados.

3.2 DESCRIPCIÓN DETALLADA DE LA ARQUITECTURA TECNOLÓGICA

En esta subsección, se detalla la arquitectura tecnológica seleccionada en base a los estudios realizados y las necesidades del sistema identificadas durante el análisis previo. La arquitectura elegida es una arquitectura distribuida, diseñada para maximizar la eficiencia, escalabilidad y fiabilidad del sistema. Está organizada en capas, lo que permite una gestión modular y clara de las diferentes funcionalidades del sistema. Las capas principales de la arquitectura son:

1. **Capa de Recolección y transformación:**
    - En esta capa, se centra en la recopilación, la transformación y estandarización de datos provenientes de diversas fuentes y en diversos formatos.
2. **Capa de Almacenamiento:**
    - La capa de almacenamiento se encarga de gestionar el almacenamiento seguro y eficiente de los datos adquiridos.

3. **Capa de Visualización:**
    - Esta capa se centra en la exploración y generación de múltiples visualizaciones de la información almacenada.

3.2.1 Selección de las tecnologías

Cómo punto de partida para el almacenamiento central y estandarizado de los datos estaremos utilizando la plataforma **Kunna** desarrollada por la Universidad de Alicante y asociada al proyecto SMARTUNI al que colabora la Universidad de Oviedo.

En cuanto al resto de tecnologías en lugar de desarrollar una nueva herramienta desde cero, optamos por desplegar una solución consolidada como **Apache NiFi** por sus robustas capacidades en la gestión de flujos de datos. Ampliamente reconocido en el ámbito de la tecnología de datos, NiFi se destaca por su enfoque en el procesamiento basado en flujos y su eficiencia al manejar una variedad de fuentes y formatos de datos.

Para nuestros objetivos de estudio, NiFi facilitará la recolección y transformación de datos desde diversas fuentes integradas en nuestro sistema diseñado, las características y capacidades de NiFi cumplen perfectamente con nuestros requisitos para la recolección y transformación eficientes de datos.

La elección de NiFi para nuestro proyecto se basa en varias ventajas clave. Primero, siendo una herramienta de código abierto, es accesible sin costos adicionales y nos ofrece la flexibilidad de personalizar y adaptar sus funciones según nuestras necesidades específicas. Además, su alta configurabilidad nos permite ajustar los flujos de datos y los procesos de transformación de manera precisa, adaptándolos a los requisitos cambiantes del proyecto y del entorno.

NiFi también destaca por su capacidad de escalabilidad. Diseñado para manejar grandes volúmenes de datos y procesos simultáneos, puede escalar horizontalmente conforme crecen las necesidades, manteniendo un rendimiento óptimo y una gestión eficiente de recursos. El almacenamiento en búfer integrado en NiFi juega un papel crucial al asegurar que no se pierdan datos en caso de interrupciones temporales o picos de carga.

Otro beneficio significativo son los numerosos procesadores que NiFi ofrece, permitiéndonos conectar con una amplia gama de sistemas y herramientas de terceros. Estos procesadores facilitan innumerables transformaciones de datos, desde formatos simples hasta complejas manipulaciones, garantizando que podamos integrar y procesar datos de manera efectiva sin importar su origen o formato.

Una característica destacada es su capacidad de rastreo y linaje de datos, permitiéndonos seguir y visualizar el camino de los datos a través del flujo de procesamiento. Esto es fundamental para auditorías, trazabilidad y resolución eficiente de problemas, facilitando la identificación rápida de errores y áreas de optimización.

Por otro lado, la visualización de datos es también un aspecto fundamental para comprender y extraer información significativa a partir de los conjuntos de datos procesados. En nuestro proyecto, tras la fase de recolección y almacenamiento de datos, surge la necesidad de una herramienta potente y flexible para la visualización de la información.

Después de evaluar cuidadosamente diversas herramientas, hemos optado por el componente **Kibana** de **ELK** como la solución óptima para nuestro proyecto. Kibana es una solución de análisis y visualización de código abierto que destaca por su integración estrecha con Elasticsearch, lo cual facilita una exploración avanzada de datos y un monitoreo en tiempo real efectivo. Esta herramienta es altamente configurable y adaptable a las necesidades específicas del proyecto. Con su interfaz de usuario intuitiva y su capacidad para crear paneles personalizados cuenta con una variedad de gráficos y mapas geoespaciales.

Kibana permite a los usuarios diseñar visualizaciones a medida que se ajusten a los requisitos del análisis de datos, incluyendo un lenguaje para los usuarios expertos o arrastrar y soltar para usuarios inexpertos.

Además, cuenta con el foro de ayuda de Elastic, que es muy útil para resolver problemas de manera rápida y efectiva.

En comparación con otras herramientas, Kibana se destaca por su flexibilidad, rendimiento en tiempo real y capacidad para adaptarse a nuestras necesidades específicas, lo que nos permite tomar decisiones informadas de manera oportuna y eficiente.

Toda la infraestructura estará desplegada en contenedores Docker, lo que nos proporciona la flexibilidad y la eficiencia necesarias para implantar y escalar nuestra arquitectura de manera consistente.

La arquitectura definida queda representada en la Ilustración 4

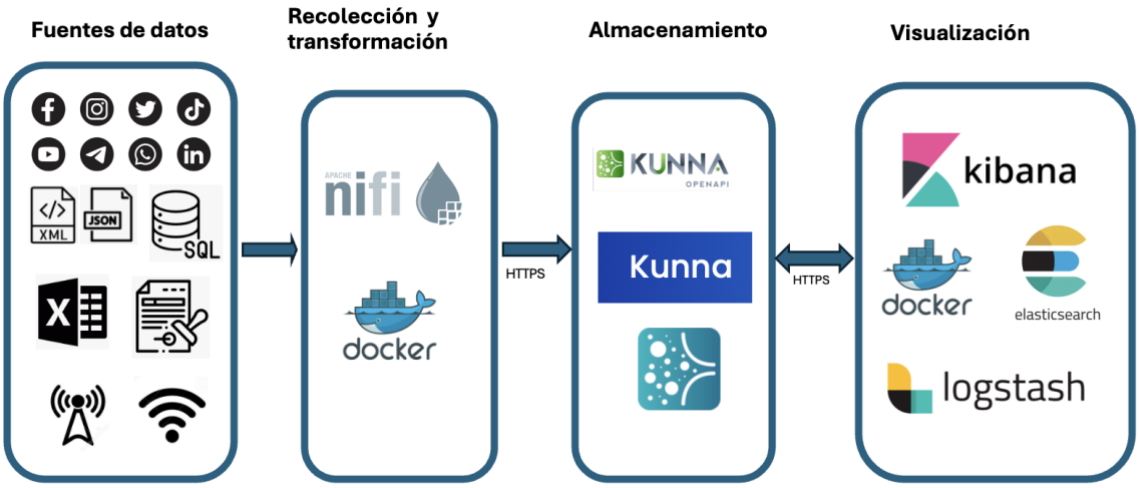


Ilustración 4. Arquitectura tecnológica

3.2.2 Estado Actual de las Tecnologías

En esta sección se presenta una revisión detallada de las características y funcionalidades de las herramientas seleccionadas.

3.2.2.1 Apache NiFi

Se basa en un software más antiguo conocido como "NiagaraFiles" creado por la Agencia de Seguridad Nacional de los Estados Unidos (NSA). Fue de código abierto como parte del Programa de Transferencia de Tecnología de la NSA y se desarrolló para abordar cuestiones relacionadas con el flujo de datos en el contexto de la seguridad, la interactividad, la tolerancia a fallos, la escalabilidad, la transformación de datos, el enrutamiento, el cifrado, la gobernanza, la procedencia, la conversión, la priorización de las colas y el almacenamiento en búfer, entre otras razones. (17)

NIFI es un sistema de flujo de datos que puede recopilar, transformar, procesar y enrutar datos. Fue construido sobre un concepto de programación basado en flujo, fue diseñado para automatizar y gestionar el flujo de datos entre sistemas.

NIFI está basado en Java y se ejecuta dentro de JVM(Java Virtual Machine) en un sistema operativo host, como se muestra en la Ilustración 5 a continuación, la arquitectura de NIFI consta de diferentes componentes, el servidor web que es responsable de alojar el comando basado en HTTP de NiFi y de permitir que el usuario acceda a NIFI a través de interfaz basada en web. Controlador de flujo que es responsable de proporcionar y programar subprocesos para su ejecución:

- Repositorio FlowFile, que es el área donde NiFi rastrea las actualizaciones de estado de los archivos de flujo.
- Repositorio de contenido que contiene el contenido de los archivos de flujo
- Repositorio de procedencia que contiene datos de eventos de procedencia.

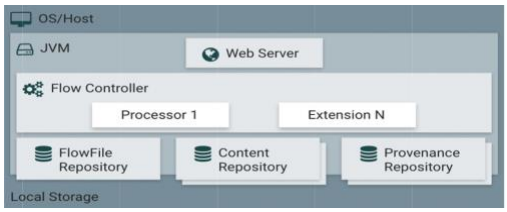


Ilustración 5. Arquitectura NIFI (Oussous, A., Benjelloun, F. Z., Lahcen, A. A., & Belfkih, S., 2018)

NiFi puede ejecutarse dentro de un clúster, cada nodo en el clúster NiFi completa las mismas tareas, pero interactúa con conjunto diferente de datos. El clúster es administrado por el coordinador del clúster, que es elegido por Apache Zookeeper.

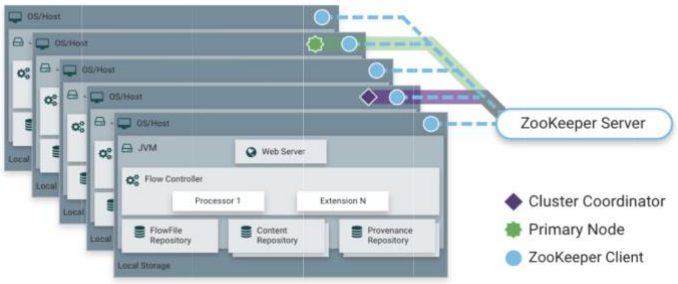


Ilustración 6. Arquitectura distribuida de NiFi

NiFi tiene una interfaz de usuario amigable basada en web que permite a los usuarios arrastrar y soltar componentes para crear el flujo de datos, los componentes se pueden iniciar y detener en tiempo real y los errores y las estadísticas se pueden ver fácilmente. NiFi almacena en búfer todos los datos en cola y permite establecer esquemas de priorización para indicar cómo se recuperarán los datos de la cola. NIFI proporciona un módulo de procedencia de datos para rastrear los datos desde el inicio del flujo hasta el final. El flujo de datos implementado es seguro ya que NiFi utiliza protocolos seguros como SSL, HTTPS, SSH y otros cifrados.

NiFi consta de una cantidad de más de 200 procesadores disponibles para leer, escribir y manipular datos. Un procesador es un elemento atómico en el flujo de datos de NiFi que puede realizar diferentes tareas, puede leer datos de múltiples recursos, enrutar, transformar y publicar datos en recursos externos. Para la ingestión de datos por lotes, los procesadores NiFi pueden leer los datos de diferentes fuentes, puede ser cualquier servidor de base de datos SQL como Oracle y MySQL, o bases de datos NoSQL como MongoDB, o extraer datos con diferentes formatos de sistemas locales o remotos.

3.2.2.1.1 Componentes básicos

A continuación, se detallan los componentes básicos de Apache Nifi:

Componente	Descripción
Flow	El workflow o topología es la definición del flujo de datos que se implementa en NiFi e indica la forma en la que se deben gestionar los datos.
Flowfile	Es el paquete de datos que viaja por el Flow entre los procesadores. Está compuesto por un puntero al propio dato útil o contenido (un array de bytes) y metadatos asociados llamados atributos. Los atributos pares clave-valor editables y NiFi los usa para enriquecer la información de provenance. Los metadatos más importantes son el identificador (uuid), el nombre del fichero (*filename*) y el path. Para acelerar el rendimiento del sistema, **el Flowfile no contiene el propio dato**, sino que apunta al dato en el almacenamiento local. Muchas de las operaciones que se realizan en NiFi no alteran el propio dato ni necesitan cargarlo en memoria. En concreto, el dato se encuentra en el llamado repositorio de contenido (*Content Repository*)
Processor	Los procesadores son los componentes principales de NiFi. Se encargan de ejecutar el proceso de extracción, transformación o carga de datos. NiFi permite realizar operaciones diversas en los processors, así como distribuir y programar su ejecución. Estos componentes también proporcionan una interfaz para acceder a los flowfiles y sus propiedades. Se pueden

	implementar nuevos processors personalizados mediante una api de programación en Java o bien usar los existentes. Los processors permiten abstraer la complejidad de la programación concurrente y pueden ejecutar en varios nodos de forma simultánea o bien en el nodo primario del clúster. Además, es posible programar su ejecución mediante *cron*, tiempo predefinido o mediante eventos de entrada. Los processors también tienen relaciones de salida (*connections*) en función de su comportamiento, por ejemplo éxito (*success*), fallo (*failure*) o reintento (*retry*). Llevan incorporado un validador de configuración y gráficas con las estadísticas de uso e indicadores de trazabilidad.

Tabla 1. Componentes básicos de Apache Nifi

3.2.2.2 Kibana en Elastick Stack

La pila ELK es un conjunto de tres herramientas: Elasticsearch, Logstash y Kibana. Aunque Logstash y Elasticsearch pueden funcionar por separado, los tres productos están diseñados para ser utilizados como una solución integrada, actualmente conocida como Elastic Stack. Un cuarto producto llamado Beats se añadió posteriormente a la pila como un cargador ligero que envía datos de los dispositivos de borde a Logstash. ELK se puede implementar en las instalaciones o utilizar como Software como Servicio (SaaS) proporcionado por una empresa externa en la nube. La pila se mantiene actualmente y se apoya activamente, bajo licencia de código abierto, por la compañía llamada Elastic. También hay un conjunto de complementos de la comunidad desarrollados por voluntarios. (23)

Logstash

Logstash es un reenvío de eventos basado en complementos con muchas características. Ingiere datos de múltiples fuentes simultáneamente, los transforma y luego los envía a otro lugar. Cada evento se procesa en una canalización de tres etapas:

entradas → filtros → salidas.

En primer lugar, hay un complemento de entrada que permite manejar un flujo de entrada específico. Logstash admite una variedad de tipos de entrada y permite capturar eventos de todas las fuentes de datos comunes, incluido el archivo CSV, el socket TCP/UDP, el punto final de la API HTTP, Elasticsearch y muchos otros.

A continuación, el evento procesado se puede mutar para alterarlo, eliminar datos prescindibles o agregar información adicional. Logstash proporciona complementos para muchas operaciones de datos diferentes. Los filtros a menudo se aplican condicionalmente y pueden realizar operaciones complejas. Por ejemplo, con el filtro Grok es posible extraer datos de un campo de cadena que contiene texto con un patrón conocido (Lista 1) y con el filtro Ruby es posible ejecutar código Ruby personalizado para el procesamiento de eventos.

Por último, al generar datos, Logstash admite una amplia gama de tipos de destino. Por lo general, todos los eventos se envían a Elasticsearch, pero Logstash también se puede utilizar de forma independiente para guardar datos en un archivo CSV, una base de datos SQL, un algoritmo de análisis de datos (es decir, Azure Machine Learning) o simplemente mostrarlo a la consola para fines de depuración. Para cada fuente de datos (es decir, un solo tipo de sensor de datos) es necesario preparar un archivo de configuración dedicado para Logstash (Lista 2). Las secciones de este archivo se refieren a las etapas particulares de la canalización de procesamiento de eventos. Las operaciones dentro de cada sección se aplican en el orden de declaración, pero el procesamiento de las secciones se realiza en un orden estricto. No es posible especificar un filtro adicional después de la sección de salida, en caso de que sea necesario crear dos eventos de salida diferentes a un solo evento de entrada, necesita duplicar el evento, agregar una marca distintiva a una de las copias del evento y procesarlo condicionalmente.

Además, para especificar cómo los datos entran y salen de Logstash, es necesario especificar el formato de estos datos. En la mayoría de los casos, se utiliza texto sin formato o JSON como códec, pero en algunos casos puede ser necesario un análisis más elaborado (es decir, contenido codificado gzip, codificación base64 o salida recopilada).

Para dividir los datos en múltiples índices de Elasticsearch, el complemento de salida de Logstash Elasticsearch acepta la especificación del patrón de índice. Un patrón de índice es una cadena con comodines opcionales que pueden coincidir con varios índices de Elasticsearch. Esto permite redirigir automáticamente el documento al índice adecuado en función de su marca de tiempo o contenido (es decir, índices divididos por día, por tipo de evento o fuente).

Elasticsearch

3.2.2.2.1 Componentes básicos

A continuación, se detallan los componentes básicos de Elasticsearch:

Componente	Descripción
Documento	Colección de campos de una determinada forma característica. Estas colecciones se encuentran definidas en JSON y son la unidad más básica de información. Además, es una información desarrollada en clave-valor.
índice	Compuesto de documentos con características parecidas.
Nodo	Es una instancia de Elasticsearch. Desempeña el cargo de almacenar lo documentos y participar en labores tanto de búsqueda cómo de indexación. No comparable a nada del mundo SQL.
Clúster	Combinación de uno o varios nodos. Garantiza la indexación grupal y búsqueda a través de todos los nodos. Cada nodo del clúster puede desempeñar una tarea o responsabilidad distinta, apareciendo los nodos de datos, maestros, de cliente o de ingesta, entre otros.
Shard	Es un fragmento de un índice. Por tanto, contendrá parte de los documentos de dicho índice. Eso sí, obviamente poseerá menos objetos en formato JSON. Un shard es una instancia de Lucene, esto consigue individualizar el motor de búsqueda para cada uno de estos mencionados fragmentos. Existen diferentes tipos de shards, cómo el shard primario (parte horizontal de un índice, siendo el "hogar" principal de un documento) o shard réplica (siguiente elemento detallado).

	No comparable a nada del mundo SQL.
Réplica	Copia del shard primario, aportando un rendimiento superior en lectura y redundancia por si aparece algún error. Las réplicas pueden darse realmente tanto en fragmentos cómo en índices.
Segmento	Un segmento es un índice invertido. Cada shard se compone de varios de estos segmentos. En la hora de una búsqueda en un shard, se hará una busca en cada uno de los segmentos al mismo tiempo. Una vez realizado este proceso, se realizará la combinación pertinente de todas estas sectorizadas búsquedas para entregar el resultado final.

Tabla 2. Componentes básicos de Elasticsearch

Elasticsearch se construyó sobre Apache Lucene, es un motor de búsqueda de código abierto, distribuible y altamente escalable que fue diseñado para tener un rendimiento óptimo. Implementa índices invertidos con transductores de estado finito para consultas de texto completo, árboles BKD para almacenar datos numéricos y geográficos, y un almacén de columnas para análisis. Tiene algunas implementaciones probadas a gran escala, como GitHub o Stack Overflow, por nombrar solo dos.

Elasticsearch tiene una API integral basada en REST que permite supervisar y controlar todos los aspectos de la configuración de un clúster. Proporciona puntos finales para realizar operaciones de creación, recuperación, actualización y eliminación (CRUD) en los datos almacenados a través de llamadas a la API HTTP. En cierta medida, Elasticsearch se puede utilizar como cualquier otro almacén de datos NoSQL. Las solicitudes de búsqueda complejas, incluidas las que implican operaciones de facetas y estadísticas, se pueden realizar en la consulta DSL (Domain Specific Language) basada en JSON (lenguaje específico del dominio). Elastic proporciona envolturas para la API en lenguajes de programación populares como Java, Python. NET, y Groovy. Además, muchos otros idiomas cuentan con el apoyo de la comunidad.

Desafortunadamente, actualizar un solo documento desde el índice de búsqueda elástica es costoso en términos de rendimiento. Si se necesita la ejecución de un gran número de operaciones de actualización, es mejor realizarlas como una sola operación de API masiva. En caso de que se requiera un mecanismo de retención de datos, se recomienda dividir los datos en múltiples índices en base a la

marca de tiempo. La operación de eliminación se realiza en todo el índice. Dependiendo del número de datos, los índices se pueden crear mensual, semanal, diario o incluso por hora.

A. función

Elasticsearch es una base de datos sin esquema, es suficiente para enviar un objeto JSON como entrada y crea un documento con la asignación adecuada para cada campo JSON automáticamente, sin sobrecarga de rendimiento. También es posible definir la asignación implícitamente, pero debe hacerse antes de agregar el primer valor de ese tipo, porque no está permitido alterar las asignaciones de campo existentes.

Elasticsearch proporciona una API dedicada para la configuración de mapeo. La asignación de tipo predeterminada tiene valores predeterminados razonables, pero cuando desea cambiar la asignación predeterminada o personalizar la indexación (almacenamiento, ignorar, finalización, etc.) debe proporcionar una definición de asignación propia. De forma predeterminada, el mapeo dinámico está habilitado. Para evitar la indexación de datos innecesariados, es posible deshabilitar la asignación dinámica para partes seleccionadas del documento; en este caso, se omitirán los campos internos sin una asignación estática definida.

Para los números enteros, lo mejor es seleccionar el tipo más pequeño posible aplicable en un caso de uso particular. Esto no reducirá el tamaño del índice, pero ayudará a que la indexación y la búsqueda sean más eficientes. También es posible asignar valores de coma flotante en enteros utilizando el factor de escala.

En algunos casos, Elasticsearch no puede deducir la asignación correcta automática; por ejemplo, el campo de cadena se puede indexar cómo campo de texto para la búsqueda de texto completo y como un campo de palabra clave para la clasificación o agregaciones. De forma predeterminada, el sistema indexará este campo en ambos sentidos. Si conoce el propósito de dicho campo, puede evitar la sobrecarga y seleccionar de antemano cómo se indexará.

También es posible que los campos de coma flotante se asignen erróneamente como enteros porque la fuente de datos envía un valor sin parte flotante cuando es cero (es decir, 123 en lugar de 123.0). En tal caso, la asignación de este campo debe estar predefinida, porque si el campo se asignará como entero y el sistema rechazará todo el registro si al menos uno de sus campos tiene un tipo incorrecto.

La asignación se puede crear junto con el índice, pero esto es un poco problemático en caso de que Logstash cree nuevos índices basados en el patrón seleccionado (es decir, un índice por día). En tal caso, es posible definir plantillas de índice que se aplicarán automáticamente cuando se creen nuevos índices. La plantilla de índice contiene la configuración del índice, las asignaciones de tipo y la base de patrones en el que el sistema decide si la plantilla debe aplicarse al nuevo índice

Kibana

Kibana fue diseñado como una plataforma de visualización para la búsqueda elástica. Proporciona una interfaz basada en la web para buscar, ver y analizar los datos almacenados en el clúster de Elasticsearch. La vista principal de Kibana se divide en 4 componentes principales: Descubrir, Visualizar, Paneles y Gestión.

La sección de gestión es donde se pueden configurar todos los componentes internos de Kibana. Es un lugar donde se deben establecer patrones de índice. Si los patrones de índice contendrán eventos de base de tiempo, también es necesario especificar la base de campo de marca de tiempo en la que Kibana ordenará y filtrará los datos. Kibana, basado en un conjunto de índices que cumplen con el patrón de índice seleccionado, muestra la lista de campos de índice junto con su tipo y propiedades. Además, es posible modificar los formateadores de campo para alterar la forma en que se muestra el valor del campo en la interfaz gráfica de usuario de Kibana.

Discover permite navegar y analizar de forma interactiva las entradas de datos puros. Los documentos de Elasticsearch se clasifican en función de los patrones de índice definidos en la sección Gestión. Se pueden buscar y filtrar fácilmente por tiempo o por propiedades del documento utilizando la sintaxis de consulta de Apache Lucene. También se puede ver el número de documentos que coinciden con la consulta de búsqueda u obtener estadísticas de valor de campo.

Debido a la arquitectura basada en complementos, Kibana se puede ampliar fácilmente para satisfacer necesidades particulares. La sección Visualizar ofrece la posibilidad de visualizar datos con uno de los complementos de visualización proporcionados. La información se puede mostrar en forma de tablas, gráficos, mapas, histogramas y muchos otros. El gran volumen se puede mostrar fácilmente en forma de gráfico circular, gráfico de barras, gráfico de líneas o gráfico de dispersión.

El panel de control permite combinar múltiples resultados guardados.

Discover y visualizar en una sola vista. Es posible organizar y cambiar el tamaño de los elementos del panel según sea necesario. Con algo de experiencia, es posible crear paneles muy sofisticados y coloridos, directamente desde la interfaz gráfica de usuario de Kibana. Los paneles se pueden guardar, compartir o incrustar fácilmente en otra página web.

Las características básicas de seguridad son gratuitas. Las características empresariales tienen licencia para uso comercial y son gratuitas para proyectos personales y no comerciales. Otra alternativa es el complemento ReadonlyREST, que ofrece un conjunto de características bastante similar bajo la licencia GPLv3.

El módulo de Vega y Vega-Lite de Kibana proporciona a los usuarios avanzados la capacidad de crear visualizaciones personalizadas utilizando gramáticas específicas. Estas gramáticas son recomendadas para aquellos que se sienten cómodos escribiendo consultas de Elasticsearch manualmente. Vega-Lite, en particular, puede ser un buen punto de partida para usuarios nuevos en estas gramáticas. Los paneles

de Vega y Vega-Lite pueden mostrar datos de múltiples fuentes, incluyendo Elasticsearch, Elastic Map Service, URL o datos estáticos, y admiten extensiones de Kibana para incrustar y agregar herramientas interactivas. Se recomienda usar Vega o Vega-Lite para crear visualizaciones que requieran agregaciones complejas, consultas con filtros de tiempo personalizados o cálculos complejos. Sin embargo, estas gramáticas tienen algunas limitaciones, como la falta de soporte para tablas y consultas condicionales. Kibana ha simplificado la escritura en JSON mediante la integración de HJSON, que permite características como cotizaciones opcionales, comillas simples o dobles, comas opcionales, comentarios y cadenas multilínea. Los usuarios pueden aprender a utilizar Vega-Lite con filtros de Kibana y datos de Elasticsearch a través de tutoriales disponibles en la documentación oficial de Kibana.

Capítulo 4 PLANIFICACIÓN Y GESTIÓN DEL TFM

En este capítulo se aborda la planificación y gestión del Trabajo. Se divide en tres secciones principales: la planificación, la ejecución y el cierre del proyecto.

4.1 PLANIFICACIÓN DEL PROYECTO

4.1.1 Identificación de Interesados

Cómo interesados en este proyecto, se identifican los siguientes:

- Los jefes y administrativos de tecnología de la Universidad son fundamentales en este proceso, ya que necesitarán una infraestructura tecnológica que maneje eficientemente esta integración de datos y que proporcione herramientas para analizar y presentar la información.
- los estudiantes, se verán beneficiados directamente por esta iniciativa, ya que la integración de datos permitirá la creación de entornos más informados y eficientes, lo que potencialmente mejorará su experiencia académica y su entorno de aprendizaje.
- los profesores, podrán utilizar los datos integrados para comprender mejor el comportamiento y las necesidades de los estudiantes, lo que les permitirá ajustar su enfoque pedagógico en consecuencia.
- los investigadores, podrán utilizar esta información para llevar a cabo análisis y descubrimientos que pueden influir en áreas de investigación específicas, como el análisis de patrones de uso, la optimización de recursos y otras áreas de interés académico y científico.

4.1.2 OBS y PBS

A continuación en Tabla 3 y Ilustración 7 se definen los OBS y PBS identificados para este proyecto. Se definen cómo diferentes perfiles los diferentes roles representados por el estudiante que desempeña durante el desarrollo del proyecto.

Perfil Profesional	id	Responsabilidad
Jefe de Proyecto	JP	Planificación, seguimiento y presentación del proyecto, asegurar que se cumplan los objetivos específicos, los hitos

		principales y el presupuesto, y la coordinación de reuniones de avance y la entrega final del proyecto.
Desarrollador Senior	DS	Definir los requisitos, integrar y validar datos de diversas fuentes, realizar consultas avanzadas y corregir errores para garantizar la coherencia y funcionalidad del proyecto.
Desarrollador Junior	DJ	Probar la conexión con sensores y fuentes de información, ejecutar pruebas de rendimiento, procesar y estructurar datos, y crear índices en Elastic Search, asegurar el manejo adecuado de las herramientas y la plataforma utilizada.
Analista de datos	AD	Diseñar dashborad y gráficas, identificar variables de estudio, elaborar diagramas de flujo y arquitectura, y realizar análisis detallados de los requisitos de datos, selección y familiarización con las herramientas de captura y transformación de datos.
Administrador de sistemas	AS	Entregar, documentar y configurar la infraestructura tecnológica, instalar y desplegar herramientas, monitorear la implementación

	operativa, y gestionar la conexión con sensores y fuentes de datos.

Tabla 3. Perfiles Profesionales

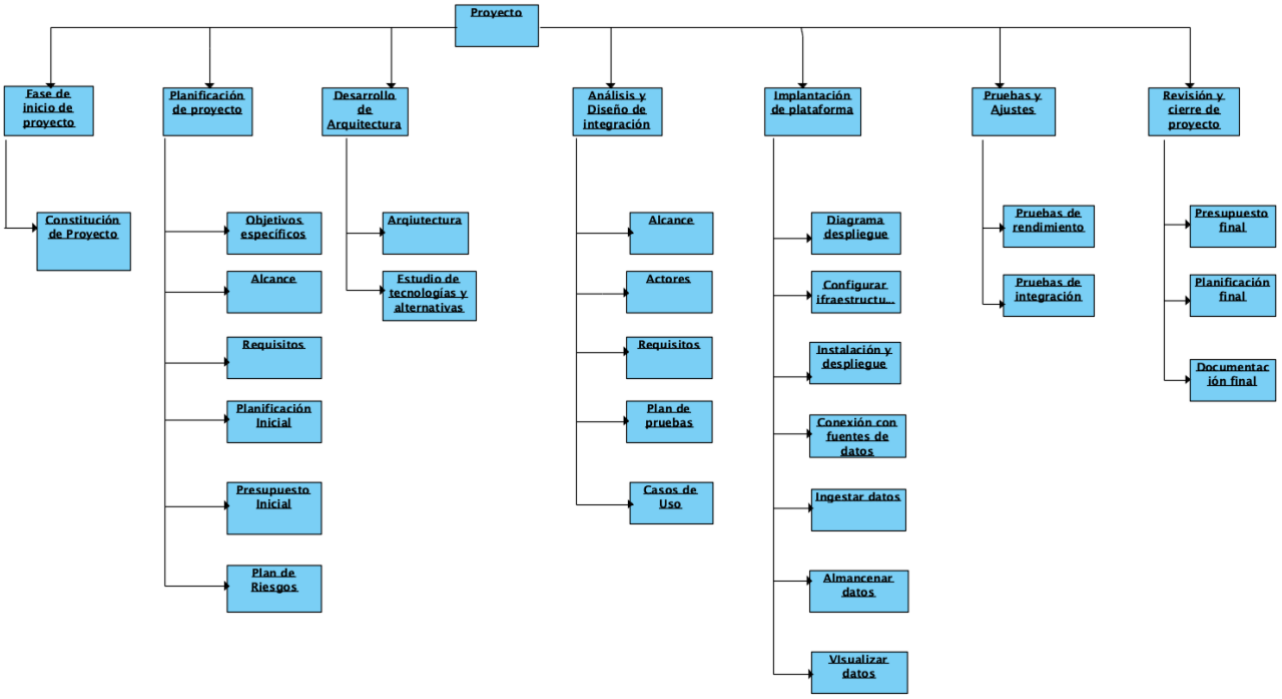


Ilustración 7. PBS

4.1.3 Planificación Inicial. WBS

En esta sección se presenta la planificación detallada del proyecto, estructurada a través de la Work Breakdown Structure (WBS). La WBS desglosa el proyecto en tareas y entregables específicos, proporcionando una visión clara de las actividades necesarias para alcanzar los objetivos establecidos. Como se observa en la planificación inicial, el proyecto dura 892 horas, con un calendario base de 8 horas diarias. Este se divide en diferentes bloques, como se detalla en Tabla 4 .

Tarea	**Horas**
Inicio de proyecto	32
Planificación de proyecto	96

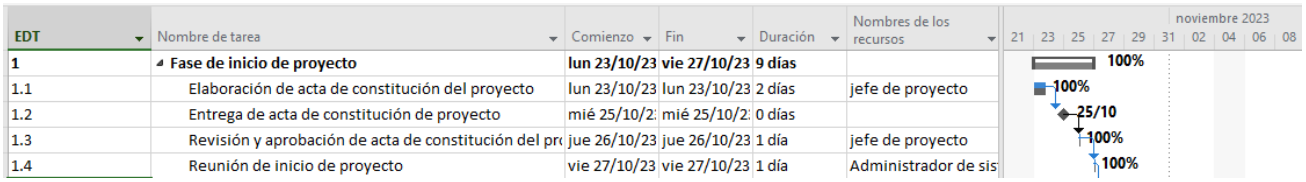
Desarrollo de arquitectura	268
Análisis y diseño de la integración	88
Implantación de la plataforma	296
Pruebas y ajustes	72
Revisión y cierre	40

Tabla 4. Resumen de tareas

Cada una de estas tareas se desglosará más a fondo en los diagramas de Gantt individuales que se presentarán a continuación.

4.1.3.1 Fase de Inicio del Proyecto

Esta fase establece los fundamentos del proyecto, incluyendo la elaboración y revisión del acta de constitución, así como la reunión inicial con todas las partes interesadas para alinear expectativas y objetivos.

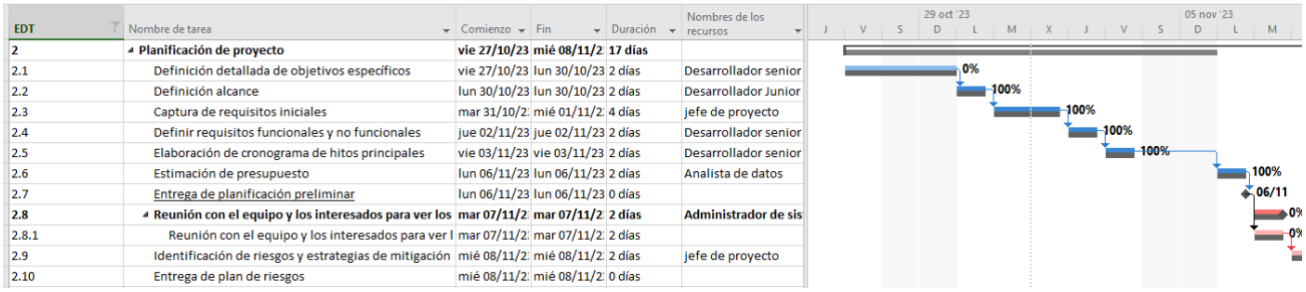


EDT	Nombre de tarea	Comienzo	Fin	Duración	Nombres de los recursos
1	**Fase de inicio de proyecto**	**lun 23/10/23**	**vie 27/10/23**	**9 días**	
1.1	Elaboración de acta de constitución del proyecto	lun 23/10/23	lun 23/10/23	2 días	jefe de proyecto
1.2	Entrega de acta de constitución de proyecto	mié 25/10/2	mié 25/10/2	0 días	
1.3	Revisión y aprobación de acta de constitución del pr	jue 26/10/23	jue 26/10/23	1 día	jefe de proyecto
1.4	Reunión de inicio de proyecto	vie 27/10/23	vie 27/10/23	1 día	Administrador de sis

Ilustración 8. Planificación de Inicio del Proyecto

4.1.3.2 Planificación y Definición de Objetivos

En esta etapa se definen detalladamente los objetivos específicos del proyecto, se establece el alcance y se planifica el cronograma y presupuesto inicial, asegurando una base sólida para la ejecución y seguimiento del proyecto

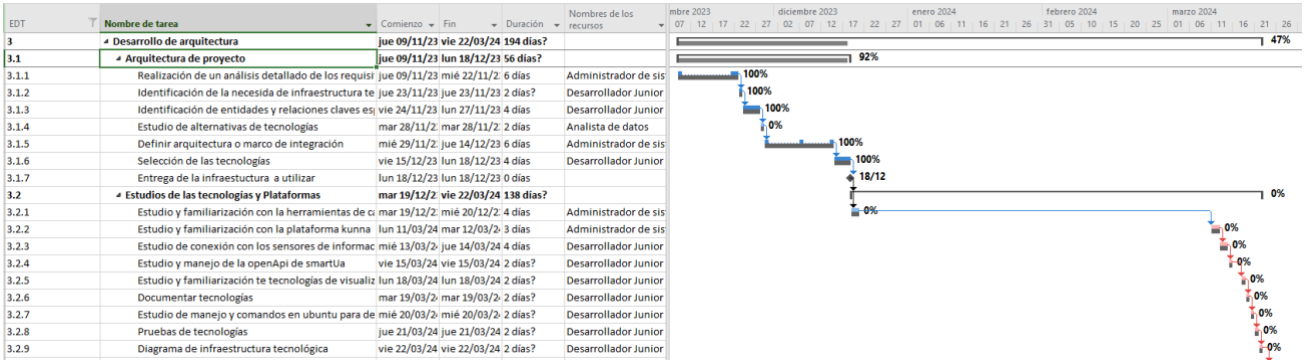


EDT	Nombre de tarea	Comienzo	Fin	Duración	Nombres de los recursos
2	**Planificación de proyecto**	**vie 27/10/23**	**mié 08/11/2**	**17 días**	
2.1	Definición detallada de objetivos específicos	vie 27/10/23	lun 30/10/23	2 días	Desarrollador senior
2.2	Definición alcance	lun 30/10/23	lun 30/10/23	2 días	Desarrollador Junior
2.3	Captura de requisitos iniciales	mar 31/10/2	mié 01/11/2	4 días	jefe de proyecto
2.4	Definir requisitos funcionales y no funcionales	jue 02/11/23	jue 02/11/23	2 días	Desarrollador senior
2.5	Elaboración de cronograma de hitos principales	vie 03/11/23	vie 03/11/23	2 días	Desarrollador senior
2.6	Estimación de presupuesto	lun 06/11/23	lun 06/11/23	2 días	Analista de datos
2.7	Entrega de planificación preliminar	lun 06/11/23	lun 06/11/23	0 días	
2.8	**Reunión con el equipo y los interesados para ver los**	**mar 07/11/2**	**mar 07/11/2**	**2 días**	**Administrador de sis**
2.8.1	Reunión con el equipo y los interesados para ver l	mar 07/11/2	mar 07/11/2	2 días	
2.9	Identificación de riesgos y estrategias de mitigación	mié 08/11/2	mié 08/11/2	2 días	jefe de proyecto
2.10	Entrega de plan de riesgos	mié 08/11/2	mié 08/11/2	0 días	

Ilustración 9. Planificación y Definición de Objetivos

4.1.3.3 Desarrollo de Arquitectura y Tecnología

Se concentra en la investigación y selección de tecnologías adecuadas, así como en el diseño detallado de la arquitectura tecnológica necesaria para soportar las operaciones del proyecto.

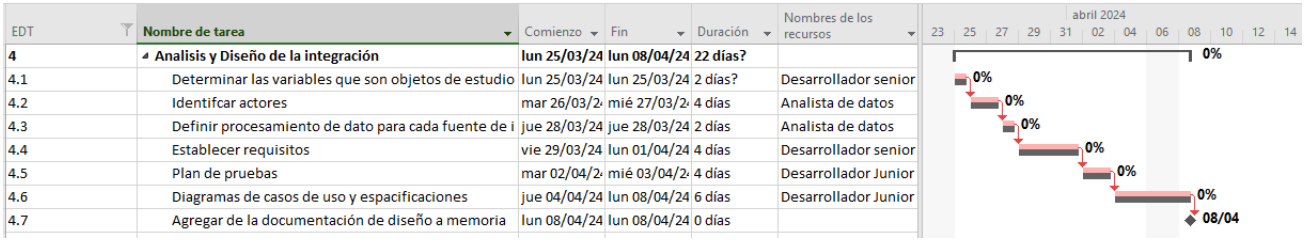


EDT	Nombre de tarea	Comienzo	Fin	Duración	Nombres de los recursos
3	Desarrollo de arquitectura	jue 09/11/23	vie 22/03/24	194 días?	
3.1	Arquitectura de proyecto	jue 09/11/23	lun 18/12/23	56 días?	
3.1.1	Realización de un análisis detallado de los requisi	jue 09/11/23	mié 22/11/2	6 días	Administrador de sis
3.1.2	Identificación de la necesida de infraestructura te	jue 23/11/23	jue 23/11/23	2 días?	Desarrollador Junior
3.1.3	Identificación de entidades y relaciones claves es	vie 24/11/23	lun 27/11/23	4 días	Desarrollador Junior
3.1.4	Estudio de alternativas de tecnologías	mar 28/11/2	mar 28/11/2	2 días	Analista de datos
3.1.5	Definir arquitectura o marco de integración	mié 29/11/2	jue 14/12/23	6 días	Administrador de sis
3.1.6	Selección de las tecnologías	vie 15/12/23	lun 18/12/23	4 días	Desarrollador Junior
3.1.7	Entrega de la infraestuctura a utilizar	lun 18/12/23	lun 18/12/23	0 días	
3.2	Estudios de las tecnologías y Plataformas	mar 19/12/2	vie 22/03/24	138 días?	
3.2.1	Estudio y familiarización con la herramientas de c	mar 19/12/2	mié 20/12/2	4 días	Administrador de sis
3.2.2	Estudio y familiarización con la plataforma kunna	lun 11/03/24	mar 12/03/2	3 días	Administrador de sis
3.2.3	Estudio de conexión con los sensores de informac	mié 13/03/2	jue 14/03/24	4 días	Desarrollador Junior
3.2.4	Estudio y manejo de la openApi de smartUa	vie 15/03/24	vie 15/03/24	2 días?	Desarrollador Junior
3.2.5	Estudio y familiarización te tecnologías de visualiz	lun 18/03/24	lun 18/03/24	2 días?	Desarrollador Junior
3.2.6	Documentar tecnologías	mar 19/03/2	mar 19/03/2	2 días?	Desarrollador Junior
3.2.7	Estudio de manejo y comandos en ubuntu para de	mié 20/03/2	mié 20/03/2	2 días?	Desarrollador Junior
3.2.8	Pruebas de tecnologías	jue 21/03/24	jue 21/03/24	2 días?	Desarrollador Junior
3.2.9	Diagrama de infraestructura tecnológica	vie 22/03/24	vie 22/03/24	2 días?	Desarrollador Junior

Ilustración 10. Planificación de Desarrollo de Arquitectura y Tecnología

4.1.3.4 Análisis y diseño de la integración

Este bloque se centra en la configuración e instalación de la infraestructura requerida, además de la realización de pruebas para asegurar su funcionamiento óptimo antes de la implementación completa.



EDT	Nombre de tarea	Comienzo	Fin	Duración	Nombres de los recursos
4	Analisis y Diseño de la integración	lun 25/03/24	lun 08/04/24	22 días?	
4.1	Determinar las variables que son objetos de estudio	lun 25/03/24	lun 25/03/24	2 días?	Desarrollador senior
4.2	Identifcar actores	mar 26/03/2	mié 27/03/2	4 días	Analista de datos
4.3	Definir procesamiento de dato para cada fuente de i	jue 28/03/24	jue 28/03/24	2 días	Analista de datos
4.4	Establecer requisitos	vie 29/03/24	lun 01/04/24	4 días	Desarrollador senior
4.5	Plan de pruebas	mar 02/04/2	mié 03/04/2	4 días	Desarrollador Junior
4.6	Diagramas de casos de uso y espacificaciones	jue 04/04/24	lun 08/04/24	6 días	Desarrollador Junior
4.7	Agregar de la documentación de diseño a memoria	lun 08/04/24	lun 08/04/24	0 días	

Ilustración 11. Planificación de Análisis y diseño de la integración

4.1.3.5 Implantación de la plataforma

Aquí se realizan las actividades relacionadas con la ingestión de datos desde múltiples fuentes, su procesamiento, almacenamiento y finalmente la visualización mediante dashboard y herramientas de análisis.

EDT	Nombre de tarea	Comienzo	Fin	Duración	Nombres de los recursos
5	Implantación de la plataforma	mar 09/04/24	jue 06/06/24	86 días	
5.1	Configuración de infraestrucuta	mar 09/04/24	lun 29/04/24	30 días	
5.1.1	Creación de diagramas de despliegue	mar 09/04/24	mar 23/04/24	10 días	Administrador de sis
5.1.2	Configuración de la infraestructura	mié 24/04/24	jue 25/04/24	4 días	Administrador de sis
5.1.3	Instalación y despliegue de herramientas.	vie 26/04/24	lun 29/04/24	4 días	Administrador de sis
5.2	Conexión de sensores	mar 30/04/24	vie 03/05/24	8 días	
5.2.1	Prueba de conexión con sensores y fuentes de inf	mar 30/04/24	mié 01/05/24	4 días	Desarrollador Junior
5.2.2	Prueba de obtención de los datos con herramient	jue 02/05/24	jue 02/05/24	2 días	Desarrollador Junior
5.2.3	Observación detallada de la estructura de la infro	vie 03/05/24	vie 03/05/24	2 días	Desarrollador Junior
5.3	Ingesta de datos de las diversas fuentes	lun 06/05/24	vie 10/05/24	10 días	
5.3.1	Realizar conexión con cada una de las fuentes de	lun 06/05/24	lun 06/05/24	2 días	Desarrollador Junior
5.3.2	Procesamiento de datos	mar 07/05/24	mar 07/05/24	2 días	Desarrollador Junior
5.3.3	Re-estructuración de la infromación de entrada al	mié 08/05/24	mié 08/05/24	2 días	Desarrollador Junior
5.3.4	Enriquecimiento de datos de entrada.	jue 09/05/24	jue 09/05/24	2 días	Desarrollador Junior
5.3.5	Agregación de meta-datos.	vie 10/05/24	vie 10/05/24	2 días	Desarrollador Junior
5.4	Almacenamiento de datos	lun 13/05/24	jue 06/06/24	38 días	
5.4.1	Envío de datos a la plataforma kunna	lun 13/05/24	mié 15/05/24	6 días	Desarrollador senior
5.4.2	Análisis y visualización de datos	jue 16/05/24	jue 06/06/24	32 días	
5.4.2.1	Diseñar visualización	jue 16/05/24	vie 17/05/24	4 días	Desarrollador senior
5.4.2.2	Implementar visualización	lun 20/05/24	lun 20/05/24	2 días	Desarrollador Junior
5.4.2.3	Monitorear carga de datos	mar 21/05/24	vie 24/05/24	8 días	Analista de datos
5.4.2.4	Creación de dashboard con los datos que se qui	lun 27/05/24	vie 31/05/24	10 días	Desarrollador senior
5.4.2.5		lun 03/06/24	jue 06/06/24	8 días	Desarrollador Junior
5.4.3	Integración efectiva de los datos	jue 06/06/24	jue 06/06/24	0 días	

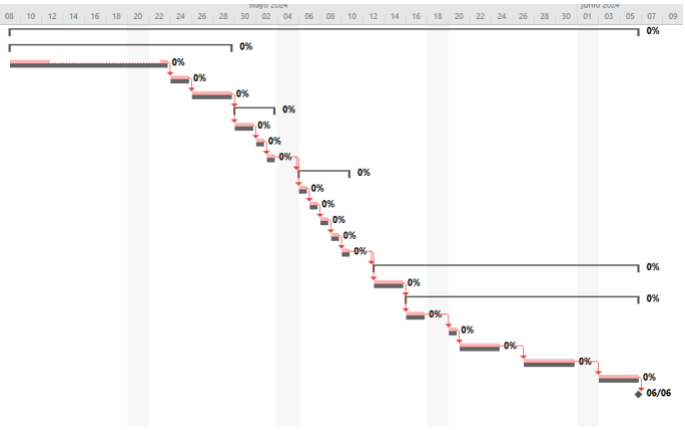


Ilustración 12. Planificación de implantación de la plataforma

4.1.3.6 Pruebas y Ajustes

Este bloque se dedica a las pruebas del sistema, desde pruebas de rendimiento hasta pruebas de integración, asegurando que el proyecto cumpla con todos los requisitos y expectativas antes de su entrega final.

EDT	Nombre de tarea	Comienzo	Fin	Duración	Nombres de los recursos
6	Pruebas y ajustes	vie 07/06/24	mié 19/06/24	18 días	
6.1	Ejecución de pruebas de rendimiento.	vie 07/06/24	mar 11/06/24	6 días	Desarrollador Junior
6.2	Identificación y corrección de errores	mié 12/06/24	mié 12/06/24	2 días	Desarrollador Junior
6.3	Validación de funcionalidad y cumplimiento de requ	jue 13/06/24	jue 13/06/24	2 días	Desarrollador senior
6.4	Integración de datos desde diversas fuentes en el er	vie 14/06/24	mar 18/06/24	6 días	Desarrollador senior
6.5	Ejecución de pruebas de extremo a extremo para ev	mié 19/06/24	mié 19/06/24	2 días	Analista de datos

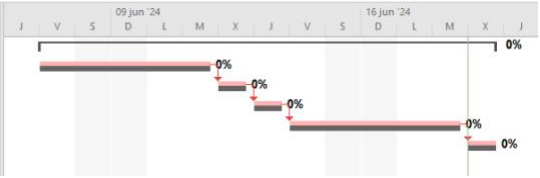


Ilustración 13. Planificación de Pruebas y ajustes

4.1.3.7 Revisión y cierre de proyecto

Este último bloque engloba las tareas referentes a la culminación del proyecto y su entrega.

EDT	Nombre de tarea	Comienzo	Fin	Duración	Nombres de los recursos
7	Revisión y cierre del proyecto	mié 19/06/24	mié 26/06/24	10 días	
7.1	Entrega de presupuesto final del proyecto	mié 19/06/24	mié 19/06/24	0 días	
7.2	Revisión de la planificación realizada	jue 20/06/24	jue 20/06/24	2 días	jefe de proyecto
7.3	Monitoreo y evaluación de la implementación opera	vie 21/06/24	vie 21/06/24	2 días	Administrador de sis
7.4	Reunión de revisión	lun 24/06/24	lun 24/06/24	2 días	jefe de proyecto
7.5	Documentación final y entrega	mar 25/06/24	mar 25/06/24	2 días	jefe de proyecto
7.6	Presentación del proyecto	mar 25/06/24	mar 25/06/24	0 días	
7.7	Evaluación de satisfacción	mié 26/06/24	mié 26/06/24	2 días	jefe de proyecto

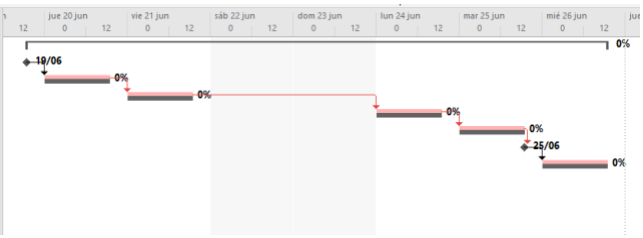


Ilustración 14. Planificación de cierre de proyecto

4.1.4 Riesgos

En esta sección se aborda la identificación y análisis de los posibles riesgos que podrían surgir durante el desarrollo del proyecto. Tanto el plan de riesgo cómo la identificación y el registro de riesgos se encuentran en los anexos “Plan de gestión de riesgos” y “Registro de riesgos”

4.1.5 Presupuesto Inicial

Aquí se presenta el presupuesto inicial del proyecto, que incluye estimaciones de los costos asociados con la ejecución del proyecto. Esto puede incluir recursos humanos, materiales, herramientas y cualquier otro costo asociado con el desarrollo del proyecto.

4.1.5.1 Presupuesto de Costes

Se presentan todos los costes identificados al proyecto. A continuación, se resumen cada una de las partidas de primer nivel del proyecto a analizar para la obtención del presupuesto de costes inicial.

Código	***Descripción***
01	**Fase de inicio de proyecto**
02	**Planificación de proyecto**
03	**Desarrollo de arquitectura**
04	**Análisis y Diseño de la integración**
05	**Implantación de la plataforma**
06	**Pruebas y ajustes**
07	**Revisión y cierre del proyecto**

Tabla 5. Resumen de partida de costes

En el anexo "Presupuesto inicial. Partidas desglosadas" se encuentra el detalle de cada partida, se estiman los recursos necesarios para cada una de las tareas que la componen (extraído de la planificación) y la valoración económica de los recursos, calculada cómo la cantidad horas del recurso por el precio del perfil.

A continuación, se muestra el presupuesto de costes agregado de las partidas anteriores mostradas del proyecto que se obtienen sumando los subtotales correspondientes a cada nivel de desglose en la jerarquía del presupuesto y se explica cómo hemos estimado lo otros costes o costes indirectos del proyecto.

	Costes agregados	
Código	***Partida***	***Total***
01	**Fase de inicio de proyecto**	**1.515,00 €**
02	**Planificación de proyecto**	**4.710,00 €**
03	**Desarrollo de arquitectura**	**9.881,25 €**
04	**Análisis y Diseño de la integración**	**3.390,00 €**
05	**Implantación de la plataforma**	**6.180,00 €**
06	**Pruebas y ajustes**	**3.840,00 €**

	Total, costes agregados	**33.446,25€**

Tabla 6. Presupuesto de costes

Para obtener el **total de costes** se tendrá en cuenta **los costes indirectos** que en este caso lo hemos estimado cómo el 15% del total obtenido en la tabla anterior (33.446,25€) y obtenemos cómo resultado el valor de **5.016,94 €**. De esta manera tenemos cómo valor para el total de costes inicial el que se muestra a continuación en la fórmula.

$$Total\ de\ costes = Costes\ indirectos + total\ Costes\ agregados$$

$$Total\ de\ costes = 33.446{,}25 + 5.016{,}94$$

$$\boldsymbol{Total\ de\ costes = 38{,}463.19}$$

Obtenemos cómo valor $\mathbf{38{,}463.19}$ **€** de total de costes.

4.2 EJECUCIÓN DEL PROYECTO

4.2.1 Plan Seguimiento de Planificación

El plan de seguimiento de la planificación del proyecto tiene como objetivo asegurar que el proyecto se desarrolla según lo planificado, identificando y gestionando cualquier desviación de manera oportuna. Este plan describe las metodologías y herramientas utilizadas para monitorear el progreso del proyecto, los indicadores clave de desempeño (KPIs) y la frecuencia de los informes de seguimiento.

Metodología de Seguimiento

Se utilizó una combinación de reuniones de seguimiento semanales, herramientas de gestión de proyectos (Microsoft Project) y revisiones mensuales del cronograma. Las reuniones frecuentes permitieron discutir el progreso, identificar y resolver problemas rápidamente, mientras que las revisiones mensuales se centraron en evaluar el avance general del proyecto.

Indicadores de Desempeño (KPIs)

Los KPIs utilizados incluyeron:

- **Cumplimiento del Cronograma**: Porcentaje de tareas completadas en comparación con las planificadas.

Teniendo en cuenta el plan de seguimiento definido, se recogen a continuación el estado del proyecto al inicio, a mediados y finales.

Este primer análisis se establece luego de haber realizado la planificación preliminar del proyecto.

Tarea	Horas	% Completado
Inicio de proyecto	32	100

Planificación de proyecto	84	87
Desarrollo de arquitectura	264	0
Análisis y diseño de la integración	88	0
Implantación de la plataforma	192	0
Pruebas y ajustes	48	0
Revisión y cierre	37,5	0

Tabla 7. Seguimiento. Inicio de proyecto

Este segundo análisis se realiza en el mes de marzo. Se estima la finalización del proyecto para el 19 de junio.

Tarea	Horas	% Completado
Inicio de proyecto	32	100
Planificación de proyecto	84	99
Desarrollo de arquitectura	272	94
Análisis y diseño de la integración	52	95
Implantación de la plataforma	192	56
Pruebas y ajustes	48	25%
Revisión y cierre	34,5	0

Tabla 8. Seguimiento Mediado de proyecto

Este tercer análisis se realiza al acabar el proyecto. El proyecto ha finalizado el 26 de junio

Tarea	Horas	% Completado
Inicio de proyecto	32	100
Planificación de proyecto	84	99

Desarrollo de arquitectura	272	99
Análisis y diseño de la integración	52	99
Implantación de la plataforma	196	100
Pruebas y ajustes	20	99
Revisión y cierre	34,5	99

Tabla 9. Seguimiento. Fin de proyecto

Análisis de seguimiento y Avance del Proyecto

Inicio del Proyecto:

- Horas totales planificadas: 745.5 horas.
- Horas ejecutadas: 116.5 horas.
- Porcentaje completado promedio: 26.6%.

Medio del Proyecto:

- Horas totales planificadas: 714.5 horas.
- Horas ejecutadas: 684.5 horas.
- Porcentaje completado promedio: 72.6%.

Final del Proyecto:

- Horas totales planificadas: 688.5 horas.
- Horas ejecutadas: 688.5 horas.
- Porcentaje completado promedio: 99.3%.

En la fase inicial, se completaron 116.5 horas de trabajo, representando un 26.6% del total. Esto incluye la finalización del "Inicio de proyecto" y un avance significativo en la "Planificación de proyecto".
En el estado intermedio, el trabajo ejecutado aumentó a 684.5 horas, que es un 95.8% del trabajo total planificado para esa fase. Las tareas críticas como "Desarrollo de arquitectura" y "Análisis y diseño de la integración" avanzaron considerablemente, y la "Implantación de la plataforma" comenzó a mostrar progreso. En la fase final, las horas ejecutadas igualaron las planificadas (688.5 horas), logrando el 100% de completitud en todas las tareas.

4.2.2 Bitácora de incidencias

La bitácora de incidencias fue una herramienta esencial para registrar y gestionar los problemas encontrados durante el proyecto, a continuación, se muestran el registro de la bitácora:

- Pérdida de permisos de acceso a TTN
- Pérdida de datos en un periodo temporal
- Saturación del almacenamiento de los contenedores de aplicaciones
- Falta de comunicación con el equipo encargado de la sensorización LoRa
- Conocimiento insuficiente sobre la filosofía de trabajo de las tecnologías utilizadas.
- Colapso de los contenedores

4.2.3 Riesgos

La siguiente Tabla 10 registra los 5 riesgos a los que se le dará seguimiento.

Identificador del riesgo	Nombre del riesgo	Descripción
2	Dificultades en la Integración Tecnológica	Incapacidad de lograr integrar la infraestructura tecnológica.
4	Falta de Acceso a Fuentes de Datos	Perdida de acceso a las Fuentes de datos a utilizar, ya sea por rotura, caída de los servicios y cualquier causa que imposibilite obtener los datos.
7	Desviación en Plazos de Entrega	Retrasos significativos en los plazos de entrega establecidos en la planificación.
8	Rotura del servidor	Rotura del servidor que aloja la infraestructura del proyecto.
18	Retrasos en la obtención de permisos de acceso	Retraso en la obtención de los permisos para acceder a la información de las Fuentes de datos, causando que otros riesgos se materialicen.

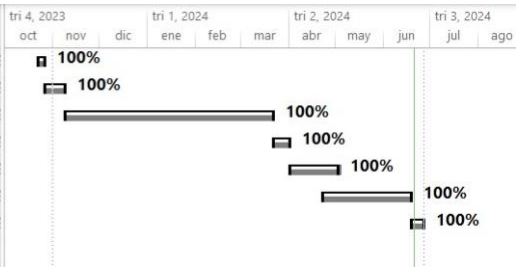
Tabla 10. Tabla de seguimiento de Riesgos

4.3 CIERRE DEL PROYECTO

4.3.1 Planificación Final

La planificación final refleja las fechas finales dedicadas al proyecto resultando en un total de 690,5 horas con una desviación de la planificación inicial de 201 horas menos de lo planificado inicialmente. La planificación de tallada se encuentra adjunta en el anexo "Planificación final"

EDT	Nombre de tarea	Comienzo	Fin	% completado
1	Fase de inicio de proyecto	lun 23/10/23	vie 27/10/23	100%
2	Planificación de proyecto	vie 27/10/23	mié 08/11/23	100%
3	Desarrollo de arquitectura	jue 09/11/23	vie 22/03/24	100%
4	Analisis y Diseño de la integración	vie 22/03/24	lun 01/04/24	100%
5	Implantación de la plataforma	mar 02/04/24	vie 03/05/24	100%
6	Pruebas y ajustes	mar 23/04/24	mar 18/06/24	100%
7	Revisión y cierre del proyecto	mar 18/06/24	mié 26/06/24	100%



4.3.2 Informe final de Riesgos

De los riesgos identificados anteriormente en la sección "Riesgos" durante el inicio del proyecto se materializaron los siguientes:

Identificador del Riesgo	Nombre	Descripción
5	Complejidad en la Implementación de visualización de datos	El proceso de implementar visualizaciones con el lenguaje Vega/Lite fue muy complejo, causando demoras en estudio.
18	Retrasos en la obtención de permisos de acceso a los servicios que se consuman.	El acceso a las fuentes de datos se perdió por cambio en los tokens de acceso y la recuperación causó perdida de información
13	Problemas con la integración de APIs externas	Problemas con la api externa de acceso a los datos almacenados, generando errores aleatorios en la recepción de los datos en la capa de visualización

4	Falta de Acceso a Fuentes de Datos	Rotura en el sensor de loRa generó que se detuviera el envío de datos y falta de acceso a los mismos.

4.3.3 Presupuesto Final

En esta sección se muestra el presupuesto final del proyecto, partiendo de lo documentado en la sección "Presupuesto Inicial Se recalcula el presupuesto de costes de la misma manera que al inicio, considerando las tareas de la "Planificación Inicial. WBS".

4.3.3.1 Presupuesto final de Costes

Se presentan todos los costes actualizados al cierre del proyecto. En el anexo "Presupuesto final. Partidas desglosadas" se encuentra el detalle de cada partida desglosada. A continuación, se resumen cada una de las partidas de primer nivel del proyecto mostrados en Tabla 5.

	Costes agregados	
Código	***Partida***	***Total***
01	**Fase de inicio de proyecto**	**1.515,00 €**
02	**Planificación de proyecto**	**4.710,00 €**
03	**Desarrollo de arquitectura**	**8.516,25 €**
04	**Análisis y Diseño de la integración**	**1.635,00 €**
05	**Implantación de la plataforma**	**4.395,00 €**
06	**Pruebas y ajustes**	**2.340,00 €**

07	**Revisión y cierre del proyecto**	**2.885,63 €**

	Total, costes agregados	**25.996,00€**

Tabla 11. Presupuesto final de costes

Para obtener el **total de costes** se tendrá en cuenta **los costes indirectos** que en este caso lo hemos estimado cómo el 15% del total obtenido en la tabla anterior (**25.996,00**€) y obtenemos cómo resultado el valor de **3.899,53 €**. De esta manera tenemos cómo valor para el total de costes inicial el que se muestra a continuación en la fórmula.

$$Total\ de\ costes = Costes\ indirectos + total\ Costes\ agregados$$

$$Total\ de\ costes = 25.996{,}00€ + 3.899{,}53€$$

$$\boldsymbol{Total\ de\ costes = 29.896{,}41}\ €$$

Obtenemos cómo valor $\mathbf{29.896{,}41}$ **€** de total de costes. Teniendo en cuenta la variación del plan de tareas entre el inicio y el fin, el presupuesto se ha reducido en aproximadamente 8.000 euros, observándose la mayor diferencia en las partidas 3, 4 y 5.

Capítulo 5 ANÁLISIS DEL SISTEMA DE INFORMACIÓN

Este capítulo se centra en el análisis detallado del sistema de información que se desarrolla

5.1 DEFINICIÓN DEL SISTEMA

5.1.1 Determinación del Alcance del Sistema

El propósito de este trabajo es establecer un marco de integración destinado a unificar diversas fuentes de datos en una única plataforma central de almacenamiento y acceso. Nuestro sistema debe tener la capacidad de recibir datos de orígenes diversos y dispares, sometiéndolos a un proceso de unificación que cumpla con requisitos de disponibilidad, tolerancia a fallos, entre otros.

Para lograr este objetivo, se realiza un estudio exhaustivo para definir una arquitectura que satisfaga los requisitos identificados del sistema. Esto incluye la selección de tecnologías adecuadas que respalden la implementación del sistema. Partiendo de la plataforma Kunna (desarrollada por la Universidad de Alicante y asociada al proyecto SMARTUNI) como base para el almacenamiento de datos, se realiza un estudio y selección de las tecnologías más adecuadas para el resto de los procesos: recolección, transformación y visualización de la información. Se diseñan e implementan los flujos de transformación necesarios para que ambas fuentes de datos cumplan con el esquema requerido.

Para probar la arquitectura propuesta, se llevará a cabo una prueba de concepto utilizando dos fuentes de datos diferentes incluyendo dispositivos inalámbricos, que pasarán por todo el flujo del sistema implementado. Los datos recopilados en la primera fase del marco serán sometidos a procesos de transformación y estandarización para convertirlos a un formato común. Una vez estandarizados, los datos se validarán para comprobar que cumplen con el esquema necesario y se almacenarán en la base de datos centralizada.

La última fase del marco definido consiste en la visualización de los datos almacenados. Para validar esta parte, se definirán escenarios de exploración de datos utilizando la gramática Vega-Lite para implementar visualizaciones. Se realizan pruebas de software para identificar errores y asegurar que independientemente de la fuente original, los datos se sometan al proceso de integración completo.

5.2 ESTABLECIMIENTO DE REQUISITOS

5.2.1 Obtención de los Requisitos del Sistema

Requisitos funcionales

- El sistema debe permitir capturar los orígenes de datos de las diversas fuentes
- El sistema debe permitir el procesamiento de los datos.
- El sistema debe validar que los datos cumplan con la estructura definida.
- El sistema debe permitir el almacenamiento de los datos
- El sistema debe permitir la exploración y creación de visualizaciones.
- El sistema debe permitir visualizar los gráficos creados, editarlos y eliminarlos
- El sistema debe permitir la gestión y control de usuario

Requisitos nos funcionales

- Alta tolerancia:

El sistema debe estar preparado para manejar volúmenes masivos de datos de manera eficiente y escalable.

- Disponibilidad:

el sistema debe estar disponible y operativo de manera continua, incluso en situaciones de fallos o mantenimiento.

- Seguridad:

Garantizar la protección de los datos.

- Escalabilidad:

Capacidad para aumentar la capacidad de procesamiento y almacenamiento según las demandas del sistema.

5.2.2 Identificación de Actores del Sistema

Se identifican lo siguientes actores:

Fuente de dato: Se entiende cómo los datos instituciones que serán incorporados al sistema y que pasan por todo el flujo definido.

Usuario Final: Es el usuario que consume de los datos expuestos en el sistema, puede ser un investigador que necesite hacer un estudio con datos o un administrador que necesite obtener cierta información brindada de los datos, etc.

Kunna: Plataforma externa utilizada para el almacenamiento central.

5.2.3 Especificación de Casos de Uso

En esta sección se expone el caso de uso para los actores definidos y el análisis de cada uno. Se muestra en la Ilustración 15 el diagrama para facilitar su comprensión.

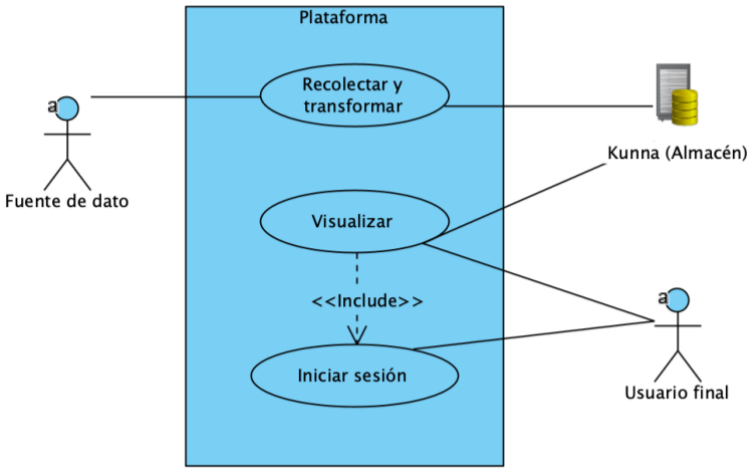


Ilustración 15. Diagrama Caso de uso

Nombre caso de uso
Recolectar y transformar
Descripción
La fuente de datos es capturada por el sistema independientemente de su origen o formato y se procesa.

Tabla 12. Caso de Uso Ingestar

Nombre caso de uso
Visualizar datos
Descripción
Los datos son visualizados a través de graficas.

Tabla 13. Caso de Uso Visualizar

Nombre caso de uso
Iniciar sesión
Descripción

El usuario debe autenticarse para poder acceder a la capa de visualización y utilizar los servicios ofrecidos.

Tabla 14. Caso de Uso Iniciar sesión

5.3 ANÁLISIS DE LOS CASOS DE USO Y ESCENARIOS

5.3.1 Recolectar y transformar

Recolectar y transformar información	
Precondiciones	-
Postcondiciones	-Los datos quedan enviados al sistema de almacenamiento
Actores	Fuente de dato
Descripción	1- La fuente de datos es capturada por el sistema independientemente de su origen o formato. 2-Los datos se transforman para cumplir con el esquema requerido 3-Los datos se validan 4- Los datos se envían a kunna
Variaciones (Escenarios secundarios)	Si hubiera algún error durante el proceso queda el registro erróneo pasa a un flujo alternativo donde se almacena y encola.
Notas	

5.3.2 Visualizar datos

Visualizar datos	
Precondiciones	Que el usuario se autentique
Postcondiciones	-
Actores	Usuario final, Kunna
Descripción	1-Se obtienen los datos almacenados en kunna y se indexan en Elasticsearch.

	Usuario final: -Crear visualizaciones -Consumir visualización -Modificar visualización
Variaciones (Escenarios secundarios)	-
Notas	-

5.3.3 Iniciar sesión

Iniciar sesión	
Precondiciones	El usuario debe acceder a la web
Postcondiciones	Acceso a los servicios de Kibana
Actores	Usuario final
Descripción	Usuario final: 1-El usuario introduce sus datos 2-El usuario presiona el botón de Log in
Variaciones (Escenarios secundarios)	
Notas	-

5.4 ESPECIFICACIÓN DEL PLAN DE PRUEBAS

A continuación, se detallará el plan de prueba para validar la arquitectura propuesta. Se tomarán como punto de partida dos tipos de pruebas: prueba de integración y prueba de rendimiento. Estas pruebas serán descritas y especificadas en los subapartados ”Especificación Técnica del Plan de Pruebas “y se mostrarán los resultados en el apartado” “Resultado de las pruebas.

5.4.1 Prueba de integración

Las prueba de integración se llevará a cabo para verificar la correcta interacción de los componentes del sistema. Esta prueba se enfoca en asegurar que los datos se integren de manera efectiva a través de las diversas tecnologías usadas, garantizando una operación fluida y coherente del sistema completo.

5.4.2 Prueba de rendimiento

Esta prueba evalúa la capacidad del sistema para manejar grandes volúmenes de datos, identificando posibles cuellos de botella y monitoreando tiempos de respuesta y uso de recursos.

Capítulo 6 DISEÑO DEL SISTEMA DE INFORMACIÓN

6.1 ARQUITECTURA DEL SISTEMA

La arquitectura del sistema propuesto se basa en un enfoque modular y escalable que permite la integración de diversas fuentes de datos en una plataforma unificada. Este diseño se estructura en tres capas principales: recolección y transformación, almacenamiento y visualización. Cada capa es fundamental en la gestión de datos, desde la recopilación inicial hasta la presentación final de la información.

Capa de Recolección y transformación

La capa de adquisición se encarga de recopilar datos de múltiples fuentes, incluyendo dispositivos IoT, sistemas existentes y otros orígenes de datos. Esta capa es responsable de la ingestión inicial de datos y su preparación para su procesamiento posterior. Se utiliza la herramienta **Apache NiFi** aprovechando las capacidades de sus procesadores integrados. Esta se ha diseñado con un enfoque en la escalabilidad y la flexibilidad. Esto permite que la infraestructura pueda crecer y adaptarse fácilmente a medida que aumenten las demandas de datos y se incorporen nuevas fuentes de información. La configuración de esta capa se diseña para establecer conexiones a través del protocolo TCP con la Capa de Almacenamiento, asegurando así un flujo eficiente y seguro de los datos recopilados.

Capa de Almacenamiento

La capa de almacenamiento se centra en el almacenamiento seguro y eficiente de los datos adquiridos en un sistema centralizado La plataforma **Kunna** es utilizada como el núcleo del sistema de almacenamiento. Kunna está diseñada para manejar grandes volúmenes de datos a través de arquitecturas distribuidas y escalables. EL acceso a esta capa será a través de **Kunna Ingest API** definido cómo un único punto de entrada para ingesta de datos de cualquier origen, siguiendo un formato universal de datos. Esta capa garantiza la integridad y la disponibilidad de los datos, permitiendo su acceso rápido y confiable en todo momento.

Capa de Visualización

La capa de visualización se encarga de presentar los datos de manera comprensible y significativa para los usuarios finales. A través de **Kunna Open API** componente de la plataforma **kunna** definido para la explotación y consumo de datos y con la utilización de **Logstash** se transfieren usando tuberías los datos almacenados en la capa de almacenamiento a **Elastic Search** para ser indexados para sus posteriores consultas y visualizaciones en **Kibana** con sus funciones de visualización de datos para

crear dashborad interactivos, informes y gráficos que permitan la exploración y el análisis de la información.

6.1.1 Diagrama de despliegue

El diagrama de despliegue de SmartUO (ver Ilustración 16) presenta una infraestructura compleja y detallada que integra múltiples componentes y plataformas para crear una solución robusta y eficiente. La **plataforma SmartUO** es el núcleo del sistema desarrollado en este trabajo, encargada de unificar diversas fuentes de datos y proporcionar una plataforma central de almacenamiento, procesamiento y acceso de la información obtenida del resto de componentes de la infraestructura. A continuación se describe en detalles cada uno de dichos componentes del diagrama y la relación entre ellos.

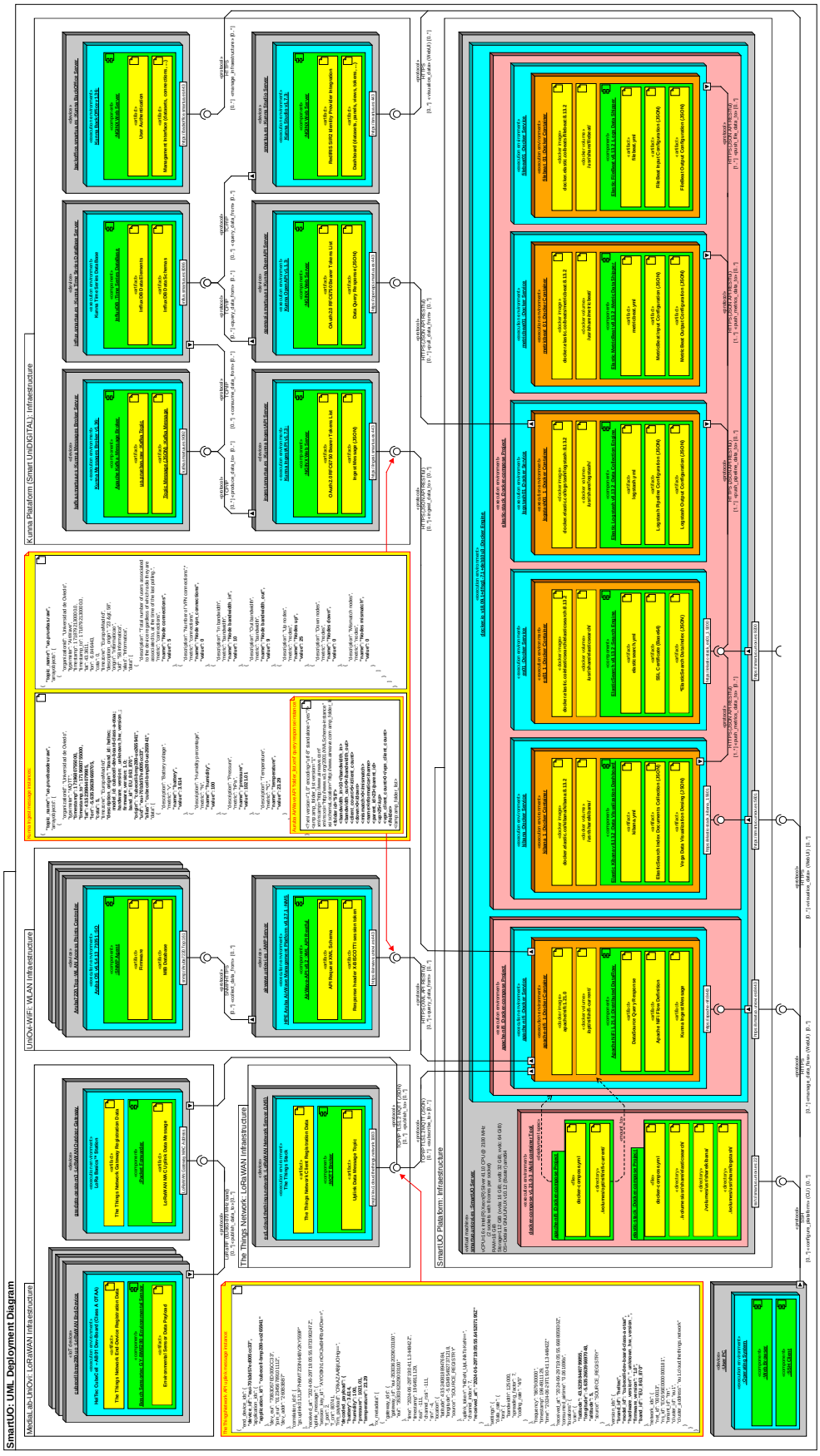


Ilustración 16. Diagrama de despliegue

A continuación se desglosan cada uno de los componentes con los que interactúa la Plataforma SmartUO:

Infraestructura MediaLab-UniOvi

En la Ilustración 17 se detalla la Infraestructura de MediaLab-UniOvi (LoRaWAN y WLAN),esta captura datos de sensores IoT desplegados en el campus, utilizando el protocolo LoRaWAN para la comunicación inalámbrica de baja potencia y largo alcance. Los datos capturados incluyen información ambiental (temperatura, humedad, calidad del aire). Estos datos son enviados mediante una comunicación LoRa a la puerta de enlace(*gateway*) que forma parte de la red LoRaWAN de TTN, siendo la encargada de publicarlo mediante el uso de protocolo MQTT en uno de los servidores de red LoRaWAN de la nube de TTN. Desde Apache NiFi nos suscribimos a la cola de mensajes MQTT publicados obteniendo los datos para su procesamiento inicial antes de integrarse en el ecosistema Kunna.

cubecell-bmp280-uo: Dispositivo final LoRaWAN que recolecta datos ambientales (temperatura, humedad, presión) utilizando el sensor Bosch Sensortec GY-BME280.

gw-dpto-oeste-m3: Gateway LoRaWAN para la transmisión de datos desde el dispositivo final a la red LoRaWAN.

The Things Network

En la Ilustración 17 se muestra la Plataforma "The Things Network":

eu1.cloud.thethings.network: Servidor de red LoRaWAN alberga un MQTT bróker que intermedia en la comunicación entre los dispositivos LoRa y Apache NiFi.

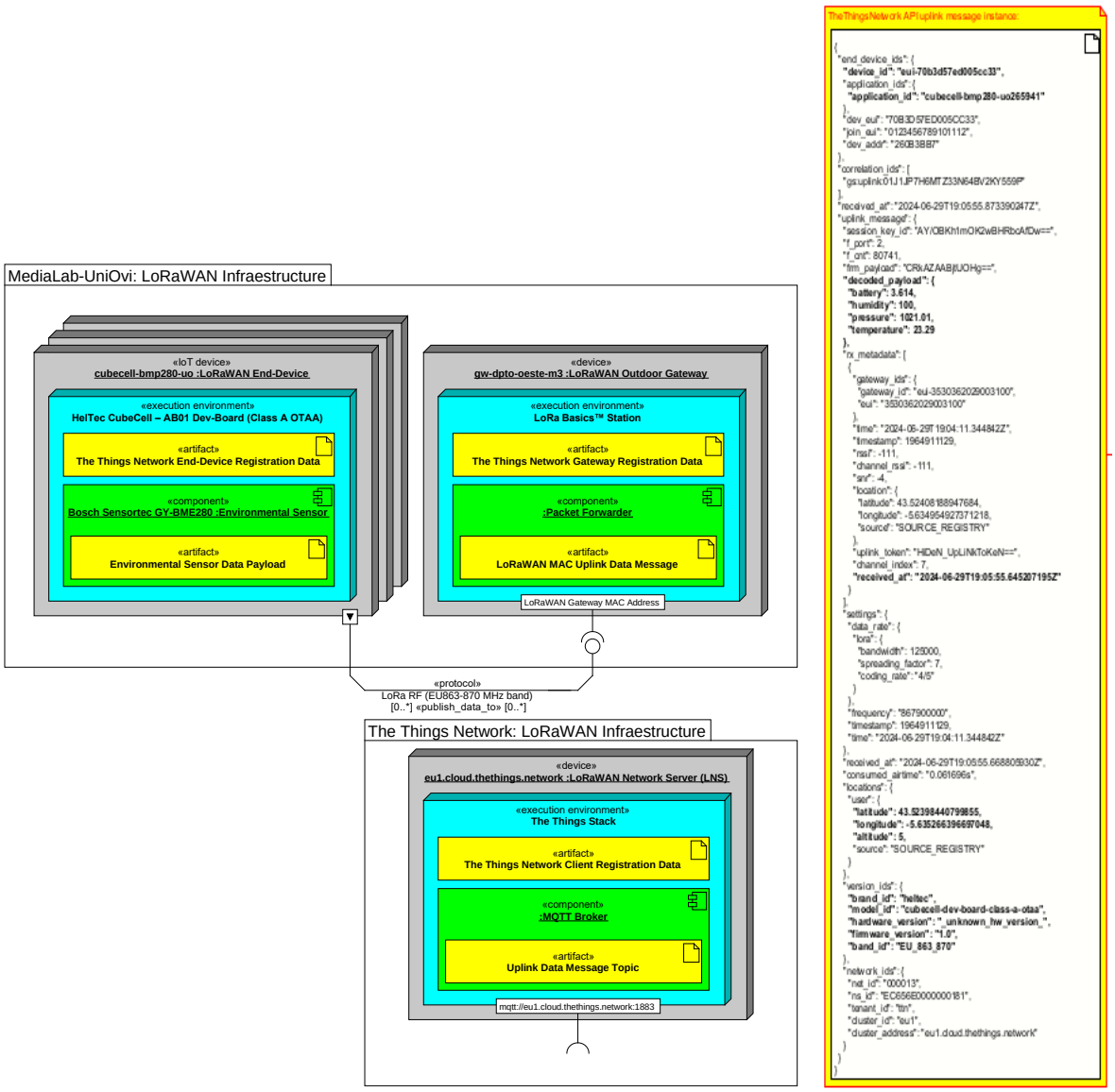


Ilustración 17. Diagrama de despliegue. MediaLab-UniOvi y TTN y una instancia de un mensaje MQTT

Infraestructura UniOvi-WiFi

En la Ilustración 18 se muestra la arquitectura de UniOvi-WiFi, encargada del monitoreo y gestión de la red inalámbrica corporativa. Basado en la tecnología del proveedor Aruba AirWave, una plataforma de gestión de red que centraliza la gestión de la infraestructura de red inalámbrica (WLAN). AirWave recopila datos sobre el rendimiento de la red, la utilización del ancho de banda, la disponibilidad de los dispositivos y el estado de los puntos de acceso. Apache NiFi se conecta a AirWave para extraer estos datos, que son transformados y enviados al Kunna IngestAPI Server.

Aruba7220.Top: Controlador de puntos de acceso WLAN.

airwave.uniovi.es: Servidor de la plataforma de gestión HPE Aruba AirWave.

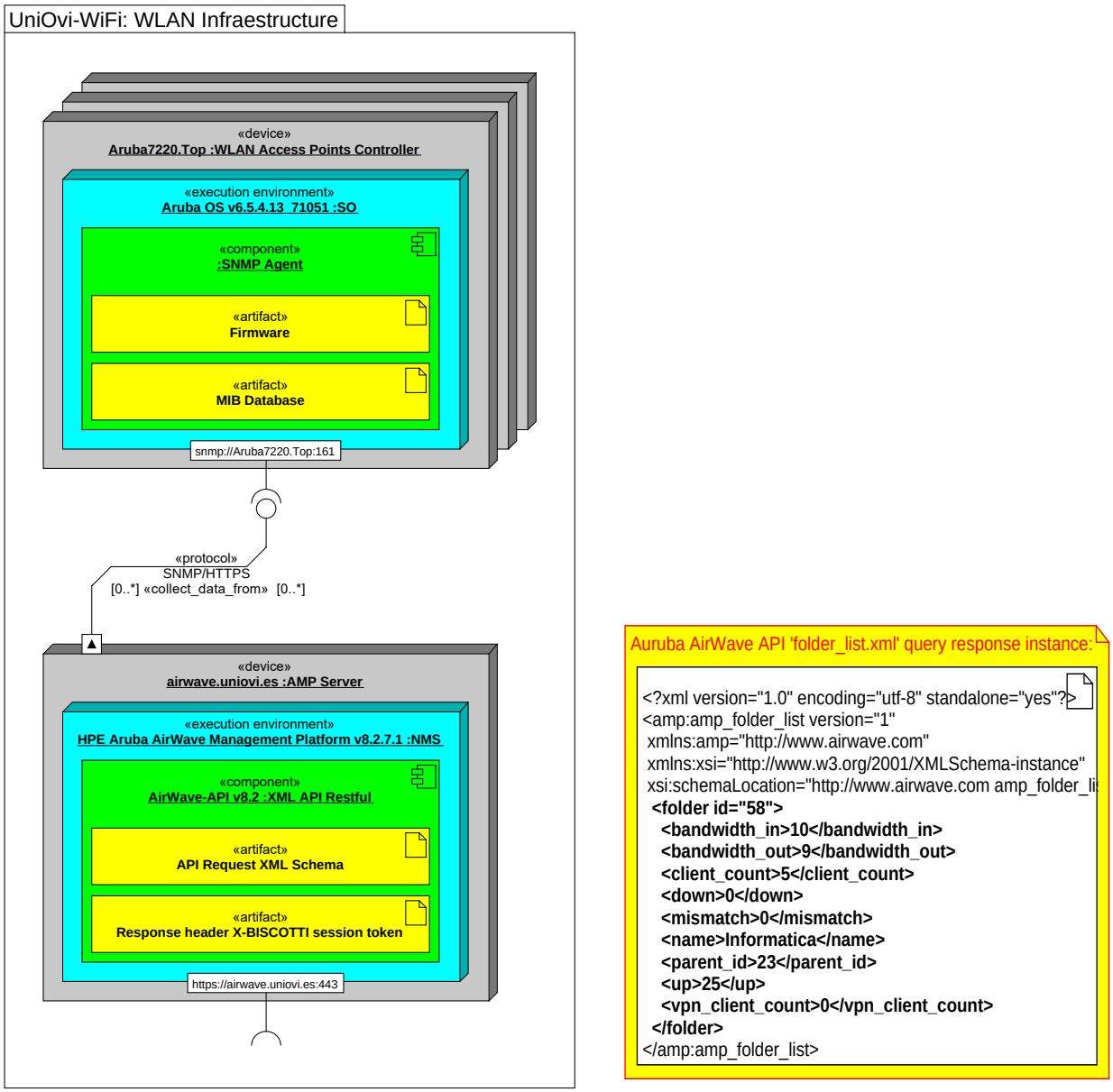


Ilustración 18. Diagrama de despliegue. UniOvi-WiFi

Plataforma Kuna

En la Ilustración 19 se muestra la infraestructura de la plataforma Kuna. En este ecosistema se integra varios servicios esenciales: Kunna BackOffice, Kunna Studio, Kunna OpenAPI, Kunna Time Series DataBase, Kunna Messages Broker y Kunna IngestAPI, es la API de ingestión de datos que recibe la información transformada desde NiFi y otros servicios, cada uno de ellos se explican en la siguiente sección de este trabajo.

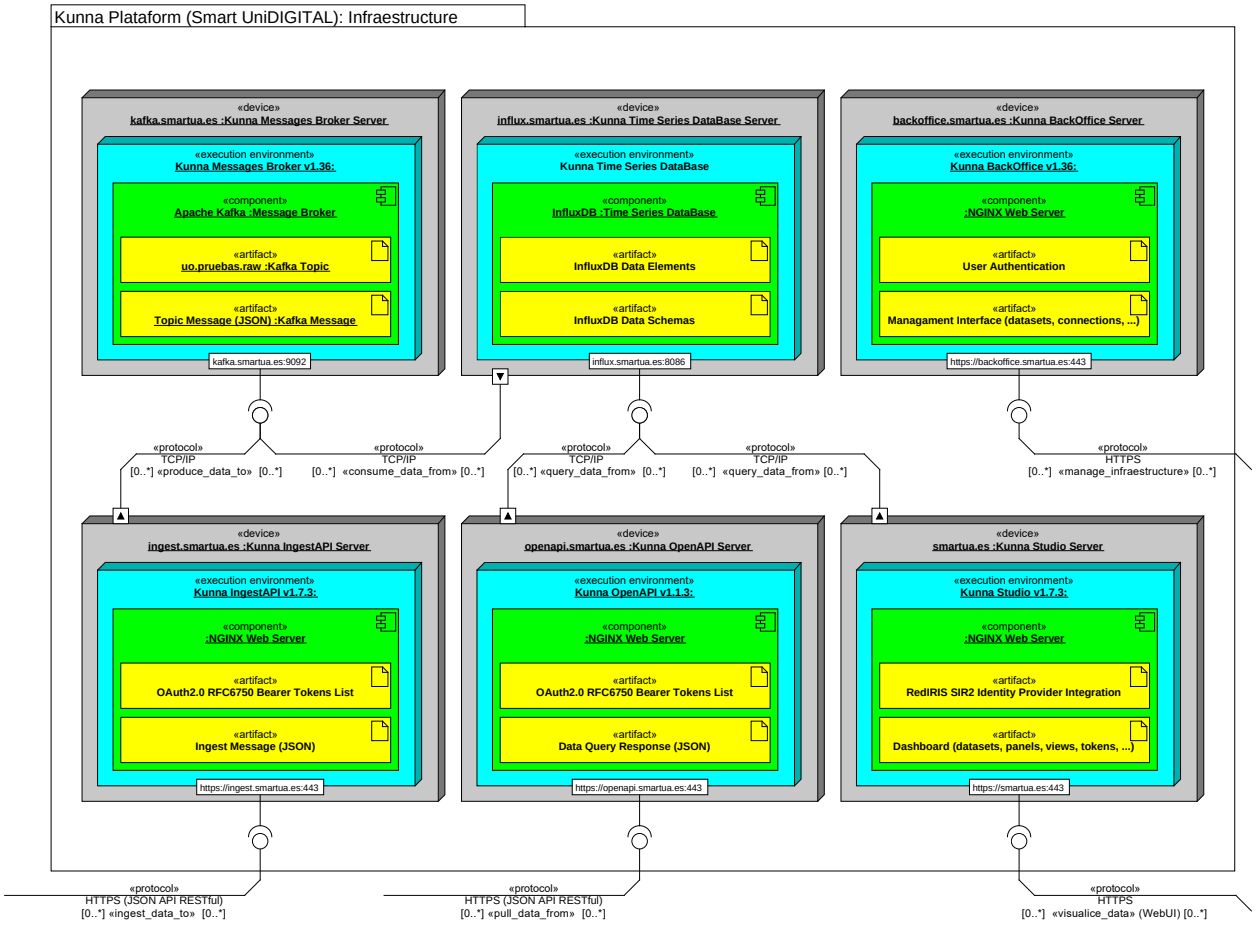


Ilustración 19. Diagrama de despliegue. Kunna Plataform

En la Ilustración 20 se muestra un ejemplo para cada fuente de datos de una instancia de mensaje de ingesta.

Kunna Ingest message instances:


```
{
   "topic_name": "uo.pruebasdev.raw",
   "arrayobjects": [
      {
         "organizationid": "Universidad de Oviedo",
         "typemeter": "MQTT",
         "timestamp": 1719680755000,
         "timestamp_to": 1719680755000,
         "lat": 43.5239844079985,
         "lon": -5.63526639669705,
         "cota": 5,
         "timezone": "Europe/Madrid",
         "description_origin": "brand_id: heltec;
               model_id: cubecell-dev-board-class-a-otaa;
               hardware_version: _unknown_hw_version_;
               firmware_version: 1.0;
               band_id": EU_863_870",
         "origin": "cubecell-bmp280-uo265941",
         "uid": "eui-70b3d57ed005cc33",
         "alias": "cubecell-bmp280-uo265941",
         "data": [
            {
               "description": "Battery voltage",
               "metric": "v",
               "name": "battery",
               "value": 3.614
            },
            {
               "description": "Humidity percentage",
               "metric": "%",
               "name": "humidity",
               "value": 100
            },
            {
               "description": "Pressure",
               "metric": "hPa",
               "name": "pressure",
               "value": 1021.01
            },
            {
               "description": "Temperature",
               "metric": "ºC",
               "name": "temperature",
               "value": 23.29
            }
         ]
      }
   ]
}
```

```
{
   "topic_name": "uo.pruebas.raw",
   "arrayobjects": [
      {
         "organizationid": "Universidad de Oviedo",
         "typemeter": "AirWave",
         "timestamp": 1718792130000.0,
         "timestamp_to": 1718792130000.0,
         "lat": 43.3611,
         "lon": -5.846443,
         "cota": 0,
         "timezone": "Europe/Madrid",
         "description_origin": "23 &gt; 58",
         "origin": "Informaticao",
         "uid": "58.Informatica",
         "alias": "Informatica",
         "data": [
            {
               "description": "Total number of users associated
               to the device regardless of which radio they are
               associated to, at the time of the last polling.",
               "metric": "connections",
               "name": "Node connections",
               "value": 5
            },
            {
               "description": "Number of VPN connections",*
               "metric": "connections",
               "name": "Node vpn_connections",
               "value": 0
            },
            {
               "description": "In bandwidth",
               "metric": "bandwidth",
               "name": "Node bandwidth_in",
               "value": 10
            },
            {
               "description": "Out bandwidth",
               "metric": "bandwidth",
               "name": "Node bandwidth_out",
               "value": 9
            },
            {
               "description": "Up nodes",
               "metric": "nodes",
               "name": "Nodes up",
               "value": 25
            },
            {
               "description": "Down nodes",
               "metric": "nodes",
               "name": "Nodes down",
               "value": 0
            },
            {
               "description": "Mismatch nodes",
               "metric": "nodes",
               "name": "Nodes mismatch",
               "value": 0
            }
         ]
      }
   ]
}
```


Ilustración 20. Diagrama de despliegue. Kunna Ingest message inastances

La **plataforma SmartUO** (ver Ilustración 21) es el núcleo del sistema desarrollado en este trabajo, encargada de unificar diversas fuentes de datos y proporcionar una plataforma central de almacenamiento, procesamiento y acceso. El servidor principal de la plataforma es una máquina virtual configurada con 16 CPUs Intel Xeon Silver 4110, 16 GiB de RAM y 112 GiB de almacenamiento distribuido en tres volúmenes. Funciona bajo el sistema operativo GNU/Linux Debian v10.12 (Buster) y utiliza Docker Engine y Docker Compose para la orquestación de contenedores. En este servidor se despliegan varios servicios críticos mediante Docker Compose que se explican a continuación:

Apache NiFi: Este componente actúa como el principal gestor de flujo de datos. Recibe datos de los sensores IoT a través de The Things Network (TTN) y de la plataforma AirWave. NiFi transforma y enruta estos datos hacia el servicio de ingesta de la plataforma Kunna.

Elasticsearch, Logstash, Kibana (ELK Stack): Elasticsearch se utiliza para el almacenamiento y búsqueda de datos. Logstash procesa los datos recibidos de la API de consumo de datos de Kunna, transformándolos según las necesidades del sistema antes de almacenarlos en Elasticsearch. Kibana proporciona una interfaz gráfica para la visualización y análisis de los datos almacenados en Elasticsearch.

Metricbeat y Filebeat: Estas herramientas se encargan de la recolección de métricas y logs del sistema, respectivamente. Metricbeat recopila métricas del rendimiento de los servicios, mientras que Filebeat envía logs a Logstash para su posterior análisis.

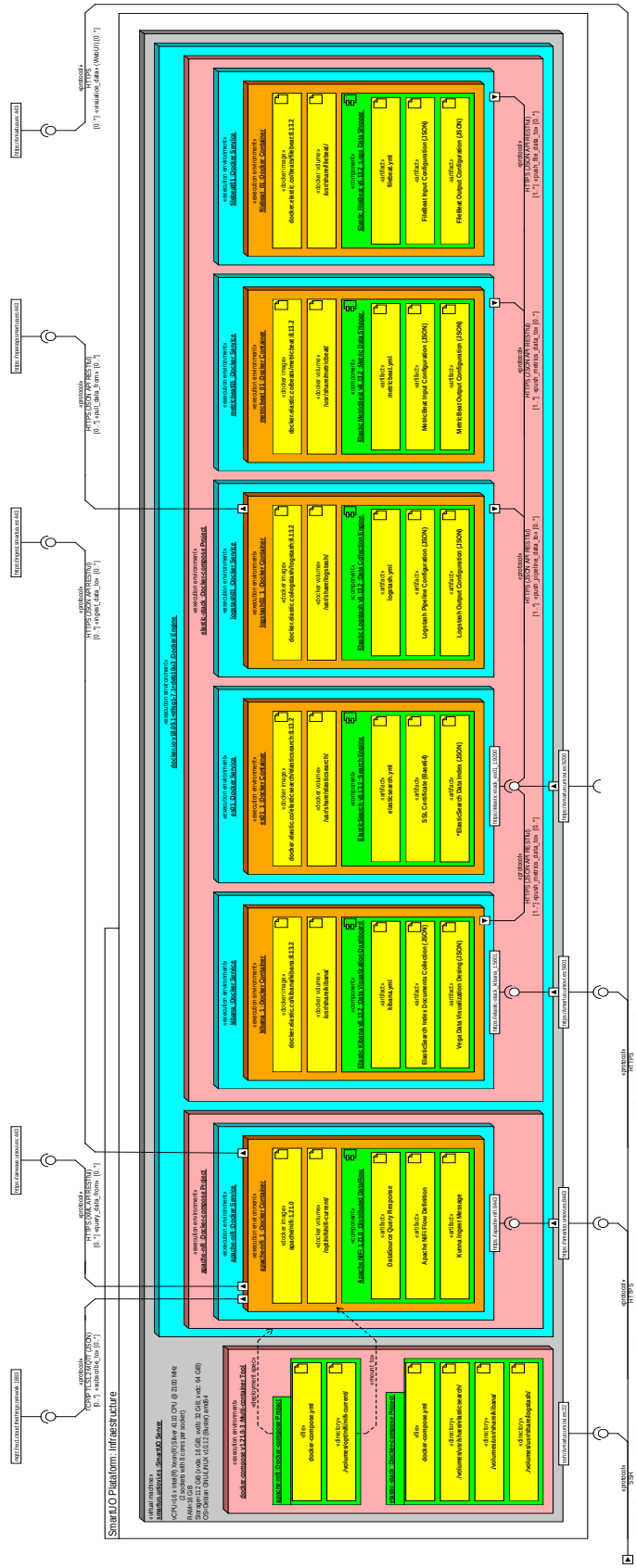


Ilustración 21. Diagrama de despliegue. SmartUO Plataform

6.2 DISEÑO FÍSICO DE DATOS

El diseño físico de datos se enfoca en la estructura de almacenamiento central de datos del sistema, definiendo cómo se organizan y gestionan los datos en la base de datos centralizada. En nuestro caso, utilizaremos la plataforma Kunna cómo base de datos principal para el sistema.

¿Qué es Kunna?

Kunna es una plataforma integral diseñada por la Universidad de Alicante para el proyecto SMARTUNI para la recopilación, estandarización, análisis y monitorización de datos. Su capacidad para unificar miles de millones de registros de diversas fuentes en un único Data HUB la convierte en una solución ideal para nuestras necesidades. Kunna está diseñada para ser escalable, tolerante a fallos, robusta, eficiente y durable.

Está pensada para trabajar tanto con datos de sensorización, dispositivos IoT y diferentes activos, siendo capaz de combinar todos ellos para generar nuevas fuentes de información. Actualmente, está formada por varios componentes entre los que destacan :

- **Kunna Studio:** Herramienta de visualización, análisis de datos, gestión de alertas y toma de decisiones a través de paneles y dashboards configurables.
- **Kunna BackOffice:** Herramienta de gestión de datasets, usuarios, permisos y conexiones al broker de datos.
- **Kunna Ingest API:** Un único punto de entrada para ingesta de datos de cualquier origen, siguiendo un formato universal de datos.
- **Kunna Open API:** API para la explotación y consumo de datos de las colecciones para aplicaciones de terceros a través del protocolo estándar HTTP.
- **Kunna BROKER:** Concentrador de datos y plataforma unificada, de alto rendimiento y de baja latencia para la manipulación en tiempo real de fuentes de datos.

Dentro de las arquitecturas para Big Data que existen en la actualidad, Kunna está diseñada para seguir una arquitectura Kappa con algunas adaptaciones. En el marco propuesto en este trabajo utilizamos la plataforma Kunna como el almacenamiento centralizado al que se enviarán las fuentes de datos integradas . Esto garantiza que los datos se persistan de manera centralizada y estandarizada, lo que facilita su posterior consumo, análisis y uso en el sistema. Además, la capacidad de Kunna para manejar grandes volúmenes de datos y su flexibilidad para la integración con otras herramientas y servicios garantizan la calidad del resultado este proyecto.

6.2.1 Descripción del SGBD Usado

El sistema de gestión de bases de datos (SGBD) que utiliza la plataforma kunna es InfluxDB. Esta herramienta es una base de datos de series temporales de alto rendimiento, diseñada específicamente para gestionar datos que cambian con el tiempo, como los datos de sensorización y los datos IoT.

Características principales de InfluxDB:

- **Arquitectura distribuida y escalable:** Permite gestionar grandes volúmenes de datos de manera eficiente y soportar un número creciente de entradas de datos y consultas sin comprometer el rendimiento.
- **Consultas eficientes en series temporales:** Ofrece capacidades avanzadas para realizar consultas y agregaciones sobre datos de series temporales, facilitando el análisis y la visualización de tendencias y patrones a lo largo del tiempo.
- **API flexible:** Proporciona una API robusta y flexible que facilita la integración con otras herramientas y servicios, lo cual es esencial para la interoperabilidad en entornos de IoT y sensorización.
- **Optimización para datos de series temporales:** InfluxDB está optimizada para la ingesta rápida de datos y para la ejecución de consultas complejas en tiempo real, lo cual es crucial para aplicaciones que requieren análisis en tiempo real y toma de decisiones basada en datos recientes.

6.2.2 Esquema de datos

Con el objetivo de estandarizar los datos, la plataforma Kunna define un esquema de datos que debe seguirse al enviar información a la capa de almacenamiento de la solución. Este esquema detalla los objetos que deben incluirse en los datos enviados.

Este esquema se detalla a continuación y se incluye en el anexo “Esquema de validación estándar”:

- topic_name: nombre del topic o colección definida donde se almacenarán los datos.
- arrayobjects: Array de objetos que se van a almacenar en la colección. [1..n]
    - organizationid: nombre de la organización
    - typemeter: nombre del medidor o sensor que va a realizar la toma
    - timestamp: fecha inicial de la toma del dato (en milisegundos)
    - timestamp_to: fecha final de la toma del dato (en milisegundos)
    - lat: Latitud donde se geoposiciona el sensor o medidor
    - lon: Longitud donde se geoposiciona el sensor o medidor

- cota: altura donde está ubicado el sensor o medidor
- timezone: Huso horario
- description_origin: descripción del dato
- origin: nombre del sensor o medidor
- uid: Identificador único global del sensor
- alias: Alias del sensor o medidor
- data: [1..n] Array de objetos con el dato o datos que el sensor o medidor x realiza la toma.
  - name: nombre de la variable medida
  - value: valor de la variable medida
  - metric: métrica en la que se mide la variable
  - description: descripción de la variable

6.3 ESPECIFICACIÓN TÉCNICA DEL PLAN DE PRUEBAS

Para validar el marco de integración diseñado primeramente se llevarán a cabo una prueba de concepto que involucra la integración completa y visualización de datos de dos fuentes diferentes.
Fuentes de datos:

- Puntos de Acceso de Wifi (uo.pruebas.raw): Estos datos, obtenidos a través de una API en formato XML, contienen métricas relevantes como el ancho de banda, las conexiones y los nodos por zonas (folders).
- Valores Obtenidos de Dispositivo Inalámbrico LoRa (uo.pruebasdev.raw): Estos datos son generados por un dispositivo inalámbrico que utiliza tecnología LoRa, conectado a TTN (The Things Network) mediante el protocolo LoRaWAN se obtienen datos en formato JSON. El sensor mide humedad, temperatura, presión y batería.

6.3.1 Prueba de Integración

La prueba de integración se enfocará en verificar la correcta interacción y flujo de datos entre los componentes del sistema. Esto incluirá la captura inicial de los datos, su transformación, almacenamiento y visualización final. Se utilizarán las dos fuentes de fuentes detallas anteriormente.
Para la prueba de integración**,** se seguirá el recorrido de un conjunto de datos para cada variable, desde su captura inicial por **Apache NiFi** hasta su visualización final en **ELK,** siguiendo el siguiente diseño:

1. **Identificación del Conjunto de Datos**: Se seleccionará un conjunto específico de datos para cada fuente, recolectando una muestra representativa en un período corto de tiempo.
2. **Flujo de Datos**: Se seguirá el recorrido de los datos desde su captura inicial hasta su visualización final en ELK, especificando cada etapa del proceso.
3. **Configuración del Entorno**: Se pondrá en pausa todo el flujo de NiFi y se configurará para ejecutar cada procesador en el modo "Ejecutar una vez". Esto permitirá observar y documentar el historial de datos (data provenance) en cada etapa.

6.3.2 Prueba de rendimiento

Para evaluar el rendimiento del sistema se integran miles de datos. Esta prueba simulará la carga de grandes volúmenes de datos para verificar la capacidad del sistema de manejar dicha carga sin comprometer su eficiencia. Durante esta prueba, se procesarán aproximadamente 200 000 mil documentos de datos desde la capa de recolección y transformación y serán enviados hasta la de almacenamiento y posteriormente a la de visualización, siguiendo el siguiente diseño:

1. **Simulación de Carga**: Se simulará la carga de grandes volúmenes de datos utilizando Apache NiFi para gestionar el flujo progresivo.
2. **Control de Flujo**: Se utilizará el control de flujo de los procesadores de NiFi para gestionar la carga y observar la Presión de Retorno (*Back Pressure*). La presión de retorno es una característica que permite a NiFi controlar el flujo de datos a través del sistema, asegurándose de que no se sobrecarguen los procesadores ni las conexiones. Hay dos tipos principales de límites de presión de retorno:
    a. **Back Pressure Object Threshold**: Número máximo de FlowFiles permitidos en una conexión**.**
    b. **Back Pressure Data Size Threshold**: Tamaño máximo de datos permitidos en una conexión.

    Cuando una de estas condiciones se cumple, NiFi aplicará presión de retorno, lo que significa que los procesadores aguas arriba (*upstream*) disminuirán o detendrán la creación o el envío de nuevos FlowFiles a la conexión saturada. Este mecanismo es crucial para mantener la estabilidad y el rendimiento del sistema bajo condiciones de alta carga.
3. **Monitorización de Recursos**: Se utilizará Metricbeat de ELK para monitorizar los tiempos de respuesta y el uso de recursos del clúster antes y después de la carga.

Capítulo 7 CONSTRUCCIÓN DEL SISTEMA DE INFORMACIÓN

En la construcción del sistema de información, se realizan actividades fundamentales que culminan en la integración efectiva de los datos. Este capítulo se organiza en diversas secciones que detallan las fases clave de este proceso.

7.1 PREPARACIÓN DEL ENTORNO DE GENERACIÓN Y CONSTRUCCIÓN

Se aborda la preparación del entorno necesario para la integración y construcción del sistema. A continuación se detallan las herramientas y programas que se utilizan.

7.1.1 Lenguajes de programación

7.1.1.1 Lenguaje de transformación para XML (XSLT)

En 1997 se propuso al W3C una especificación para un lenguaje de hojas de estilo para XML llamado eXtensible Style Sheet Language (XSL) (Adler et al., 1997). Un potente lenguaje de transformación XML: las transformaciones de lenguaje de hoja de estilo extensible (XSLT) se generaron extendiendo XSL con variables y la capacidad de pasar valores de datos entre reglas de plantilla. La especificación XSLT 1.0 fue aprobada y recomendada por el W3C en 1999 (Clarke, 1999). La función principal original de XSLT era permitir a los usuarios escribir transformaciones de XML a HTML, describiendo así la presentación de documentos XML. Hoy en día mucha gente utiliza XSLT como herramienta para transformaciones de XML a XML (Bex, Maneth y Neven, 2002).
Este lenguaje se usó para transformaciones en los datos a procesar en formato XML , se adjunta en el anexo ”XSLT”

7.1.1.2 Lenguaje de transformación para JSON(Jolt)

Jolt es un lenguaje de transformación de JSON a JSON escrito en Java donde la especificación para la transformación es en sí misma un documento JSON. Es una contribución de código abierto que se lanzó en 2013 con la licencia Apache-2.0, el proyecto está disponible en Github. Jolt surgió del proyecto API de plataforma de la empresa Baazarvoice para migrar el backend de Solr/MySql a Cassandra/Elasticsearch. Proporciona un conjunto de transformaciones que se pueden encadenar para

formar la transformación JSON general. No es compatible con otros lenguajes o plataformas distintas de Java. Jolt fue adoptado recientemente por Apache y recibió soporte en su software NiFi, donde Jolt se incluye como parte del conjunto estándar de procesadores que permite a los usuarios usar las especificaciones de Jolt para contenido de flujo de datos JSON. Jolt también pasó a ser compatible con Apache Camel 2.16.
Este lenguaje se usó para procesar los datos recibidor en formato JSON, se adjunta en el anexo "Jolt"

7.1.1.3 Ruby

Ruby es un lenguaje de programación creado por Yukihiro "Matz" Matsumoto en 1995, que se destaca por su belleza, practicidad y amenidad. Combina características de lenguajes como Perl, Smalltalk, Eiffel, Ada y Lisp, ofreciendo una mezcla equilibrada entre programación funcional e imperativa. Ruby es conocido por su filosofía de hacer la programación natural y no excesivamente simple, similar a la complejidad del cuerpo humano. Su popularidad se ha visto impulsada por el éxito del framework Ruby on Rails y su comunidad activa. Además, Ruby es completamente libre, altamente flexible y orientado a objetos, lo que facilita la legibilidad y la modificación del código. Implementaciones como JRuby, Rubinius y mruby expanden sus aplicaciones en diversos entornos, manteniendo su esencia de elegancia y eficiencia.
Este lenguaje se usó para transmitir los datos de la capa de almacenamiento a la capa de visualización, se adjunta en el anexo "Logstash pipeline"

7.1.1.4 Vega-Lite

Vega-Lite es una gramática de alto nivel para crear visualizaciones de datos interactivas de manera rápida y eficiente mediante una sintaxis JSON concisa y declarativa. Las especificaciones de Vega-Lite describen las visualizaciones como asignaciones de datos a propiedades de marcas gráficas, y su compilador produce automáticamente componentes visuales como ejes, leyendas y escalas, basándose en reglas predeterminadas que pueden ser personalizadas por el usuario. Vega-Lite admite transformaciones de datos (como agregación y filtrado) y transformaciones visuales (como apilamiento y facetado), permitiendo composiciones en capas y vistas múltiples. Su novedosa gramática de interacción permite a los usuarios definir la semántica interactiva mediante selecciones, que parametrizan las codificaciones visuales y definen el procesamiento de eventos de entrada. Esto permite descomponer el diseño de interacción en unidades semánticas concisas, ofreciendo una potente

combinación de reglas de codificación visual y álgebra de composición para crear visualizaciones interactivas complejas y personalizables.(34).
Esta gramática fue utilizada para la implementación de las visualizaciones, se adjunta en el anexo "Código de visualizaciones avanzadas"

7.1.2 Herramientas y programas usados para el desarrollo

7.1.2.1 Docker

Docker es una plataforma abierta para desarrollar, enviar y ejecutar aplicaciones. Docker permite separar aplicaciones de su infraestructura para que pueda entregar software rápidamente. Con Docker, se puede administrar infraestructura de la misma manera que se administra aplicaciones. Al aprovechar las metodologías de Docker para enviar, probar e implementar código, se puede reducir significativamente el retraso entre escribir el código y ejecutarlo en producción. Docker ha sido utilizado para desplegar y desacoplar todas las tecnologías de la implementación.

7.1.2.2 PuTTy

PuTTY es una herramienta de comunicación para ejecutar sesiones interactivas de línea de comandos en otras computadoras, generalmente a través del protocolo SSH. PuTTy ha sido utilizado para la comunicación con la máquina virtual.

7.1.2.3 Postman

Postman es una empresa de software global que ofrece una plataforma API para que los desarrolladores diseñen, creen, prueben y colaboren en API.
Postman fue utilizado durante el desarrollo para probar todas las APIs utilizadas en la implementación.

7.1.2.4 NiFi processor

Partiendo de lo estudiado en "Componentes básicos" para comprender que funciones cumple un processor de NiFi. Se han utilizado varios de los mismos para diferentes etapas de la implementación, cómo por ejemplo se ExecuteProcess y ConsumeMQTT para la captura de los datos de sus orígenes, SplitXML, TranformXML, MergeContent, JoltTransformJSON para las transformaciones realizadas

en los datos, ValidateJson para validar que los datos transformados cumplan con el estándar necesario y InvokeHTTP para el envío de los datos ya transformados y validados a la fase de almacenamiento.

7.1.2.5 Elasticksearch console

La consola Dev Tools de Elasticsearch es una herramienta integrada en Kibana que facilita la interacción con un clúster de Elasticsearch mediante una interfaz de línea de comandos basada en el navegador. Es especialmente útil para desarrolladores y administradores que necesitan realizar consultas, gestionar índices y analizar datos de manera eficiente. Esta consola ha ido utilizada durante el desarrollo para realizar consultas sobre los datos almacenados.

7.1.2.6 Custom kibana visualization

Kibana ofrece esta poderosa funcionalidad para crear visualizaciones personalizadas utilizando Vega y Vega-Lite, lenguajes que permiten la creación de gráficos y visualizaciones complejas y altamente personalizables. Esta se ha utilizado para crear las visualizaciones de la implementación.

7.1.2.7 Microsoft Office 365

Paquete ofimático utilizado para la documentación del trabajo.

- Excel para la creación de los presupuestos
- Project para la planificación del proyecto.
- Word para la memoria
- Power Point para la presentación

7.1.2.8 Visual Paradigm

Visual Paradigm es una aplicación de software diseñada para que los equipos de desarrollo de software modelen sistemas de información empresarial y gestionen procesos de desarrollo. Además del soporte de modelado, esta tecnología proporciona capacidades de generación de informes e ingeniería de código, incluida la generación de código.
Esta herramienta se usó para crear diagramas UML para la documentación del trabajo.

7.1.2.9 UMLet

UMLet es una herramienta UML gratuita y de código abierto con una interfaz de usuario sencilla. Esta herramienta se utilizó para el diagrama de despliegue de la solución

7.2 IMPLANTACIÓN Y DESPLIEGUE

En esta sección se realizará una explicación de las configuraciones iniciales para la implantación de la plataforma. Se clarificará los pasos a seguir para desplegar el escenario planteado en una máquina virtual **Debian** GNU/Linux 10. El lector encontrará instrucciones detalladas sobre la instalación y configuración de **Docker Compose**, **Apache NiFi**, y **ELK**. Se recomienda leer la sección dedicada al "Estado Actual de las Tecnologías", quedarán explicados conceptos teóricos y funcionamientos básicos necesarios para realizar una mejor comprensión.

7.2.1 Consideraciones previas

Cómo primeros pasos necesitaremos realizar la instalación de **Docker Compose** en nuestra máquina virtual (en este caso versión 1.21.0) para poder empaquetar todas las tecnologías que estaremos utilizando. Desde un terminal conectado por ssh siguiendo los siguientes pasos podemos obtener la instalación satisfactoria.

```
sudo curl -L \
"https://github.com/docker/compose/releases/download/1.21.0/docker-
compose-$(uname -s)-$(uname -m)" \
    -o /usr/local/bin/docker-compose
sudo chmod +x /usr/local/bin/docker-compose
docker-compose -version
```

Luego de la instalación crearemos un directorio para cada proyecto de Docker Compose con el comando:

```
mkdir -p SmartUO/apache-nifi/volumes
mkdir -p SmartUO/elastic-stack/volumes
```

En cada uno de estos directorios se ubica el correspondiente fichero `docker-compose.yml` donde se describe toda la información para la instalación y configuración de la que parten las instancias de las herramientas.

7.2.2 Despliegue de Apache NiFi

Para el despliegue del contenedor oficial de **Apache NiFi** con Docker Compose es necesario descargar la imagen oficial. El contenido completo del Docker Compose se encuentra disponible en el anexo "Despliegue de Apache NiFi". Después estarán las condiciones creadas para ser inicializada la instancia con el comando:

```
docker-compose up -d
```

Como comprobación de la correcta instalación, podríamos teclear la dirección en nuestro caso https://156.35.98.30:8443/nifi/ desde el navegador y poder ver lo siguiente:

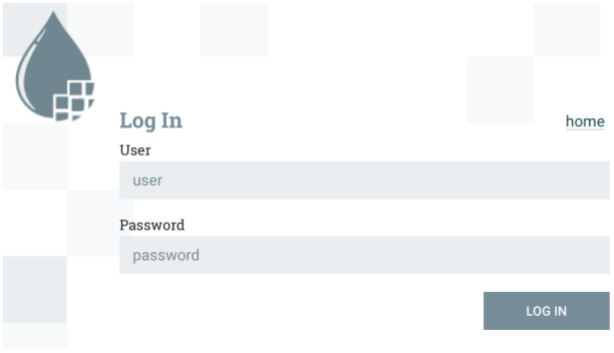


Ilustración 22.Captura de Log In de Apache NiFi

7.2.3 Despliegue de Elastic Stack

Para el despliegue del contenedor oficial de **ELK** con Docker Compose es necesario descargar la imagen oficial. El contenido completo del Docker Compose se encuentra disponible en el anexo " Despliegue de Elastick Stack".

Antes de lanzar la instancia de ELK es necesario preparar las configuraciones de **Logstash** para poder conectar la información ingestada ya en la capa de almacenamiento con Elasticsearch, para esto necesitamos crear el directorio para almacenar los ficheros de configuración de las tuberías *(pipeline)* con el siguiente comando.

```
mkdir -p SmartUO/elastic-stack/volumes/usr/share/logstash/pipeline
```

Estos fichero se adjuntan en el anexo "Logstash pipelines". Las tuberías están configuradas con parámetros que definen los datos de entrada, los cuales son cargados a través de la Open API que ofrece Kunna. Los datos procesados por Logstash se envían directamente a Elasticsearch,

insertándolos en el índice especificado. También es necesario crear el fichero `pipelines.yml` en la ruta `/volumes/usr/share/logstash/`. Este fichero contiene la configuración de las tuberías creadas para que el contenedor de Logstash pueda encontrarlas y cargarlas correctamente, el contenido del fichero se muestra a continuación.

```
- pipeline.id: pipeline-wifi
path.config: "/usr/share/logstash/pipeline/pipeline-wifi.conf"

- pipeline.id: pipeline-loRa
path.config: "/usr/share/logstash/pipeline/pipeline-loRa.conf"
```

Después estarán las condiciones creadas para ser inicializadas las instancias con el comando

```
docker-compose up -d
```

Como comprobación de la correcta instalación, podríamos teclear la dirección en nuestro caso http://156.35.98.30:5601/ desde el navegador y poder ver lo siguiente:

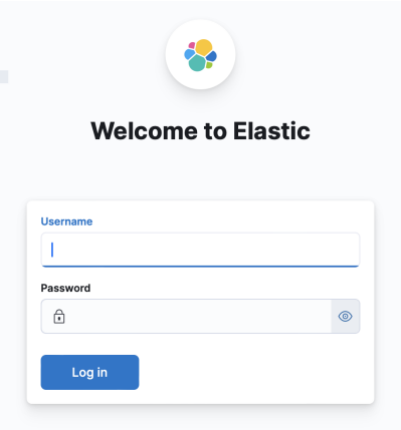


Ilustración 23. Instalación de Kibana

7.3 RESULTADO DE LAS PRUEBAS

A continuación, se mostrarán los resultados de las pruebas realizadas.

7.3.1.1 Prueba de integración

Por cuestiones de tamaño algunas trazas de la prueba realizada se encuentran en el anexo “Resultado de las pruebas”.

- **Puntos de Acceso de Wifi (uo.pruebas.raw)**

En la Ilustración 24 se muestra el flujo implementado para esta variable. Es importante destacar que daremos seguimiento a <folder id="70">, donde 'folder' se refiere a la agregación de puntos de acceso Wifi, p.ej., “ciudad”, “campus”, ”edificio”.

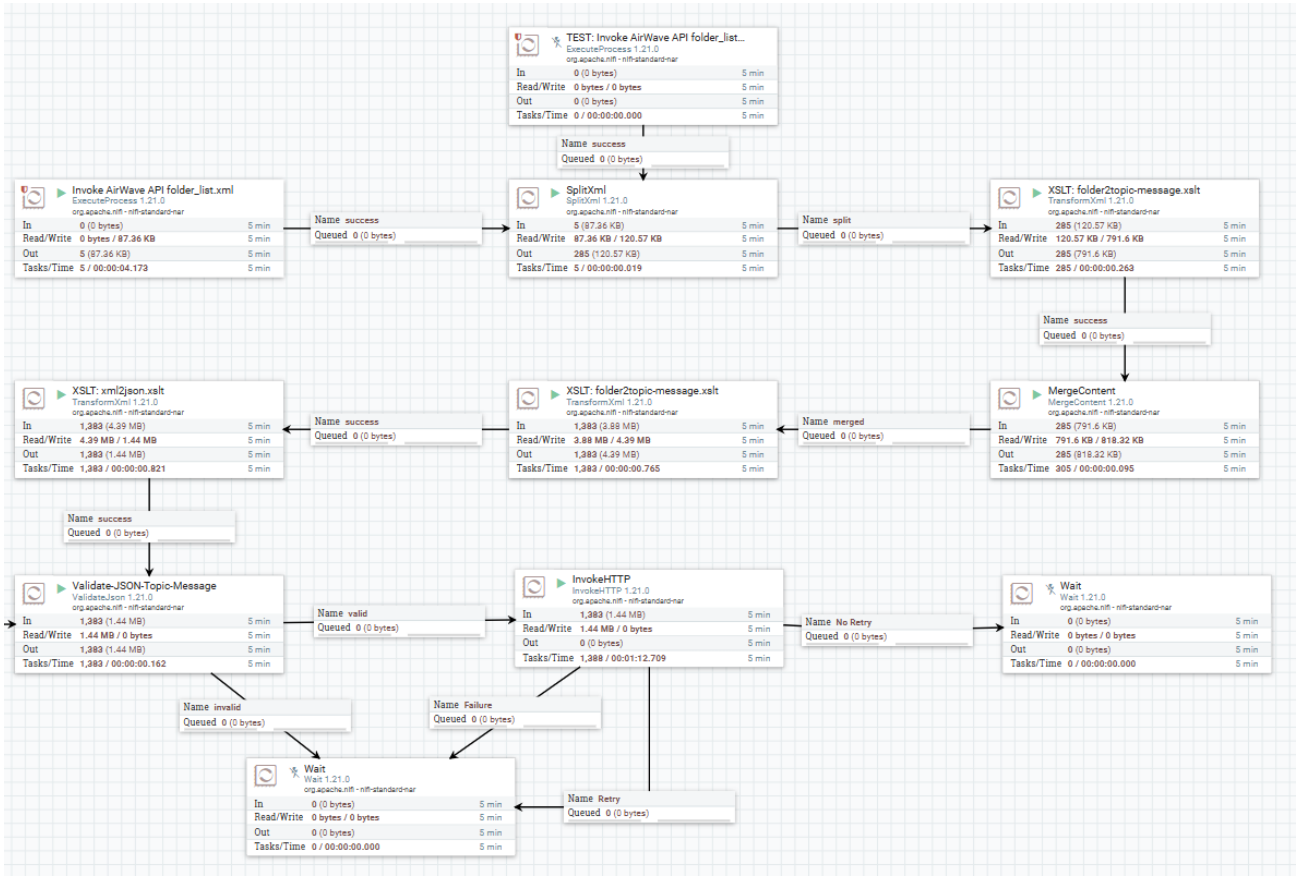


Ilustración 24. Flujo NiFi uo.pruebas.raw

En el anexo “Captura de datos” se adjunta la primera recepción de los datos, que posteriormente se transforma para ajustarse al esquema detallado anteriormente en “Esquema de datos”. El anexo “Transformación de datos” ofrece un desglose paso a paso del proceso aplicado a este conjunto de datos, primera parte del proceso de integración de los datos en el sistema.

La Ilustración 25 muestra una captura de pantalla del proceso de validación y envío exitoso de los datos a la API de ingestión del almacenamiento central de Kunna, lo que demuestra la correcta fluidez de los datos durante esta parte del proceso.

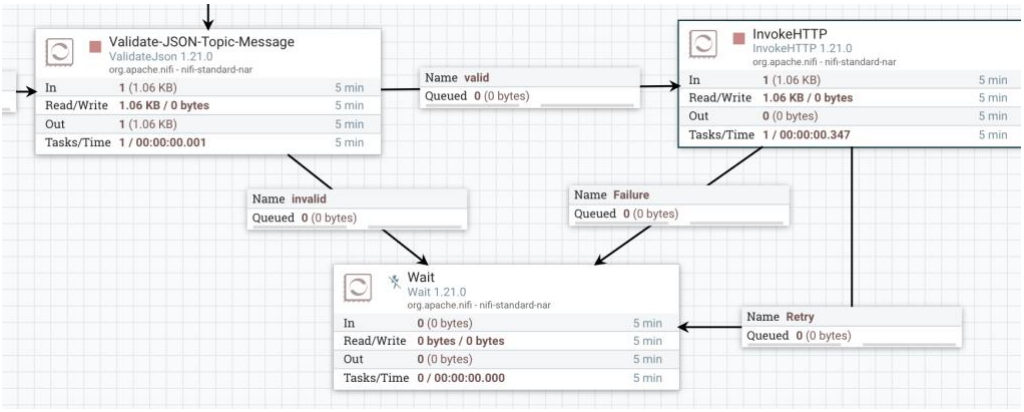


Ilustración 25. Prueba de integración wifi – validación

Luego de almacenados los datos, se pasa al proceso de indexación par la visualización, en la Ilustración 26 se observa una captura de la consola de Elasticsearch donde se muestran los datos previamente capturados procesados y almacenados. Llegado este punto los datos han pasado por cada uno de los componentes del sistema, comprobando que se integraron efectivamente cada una de las tecnologías.

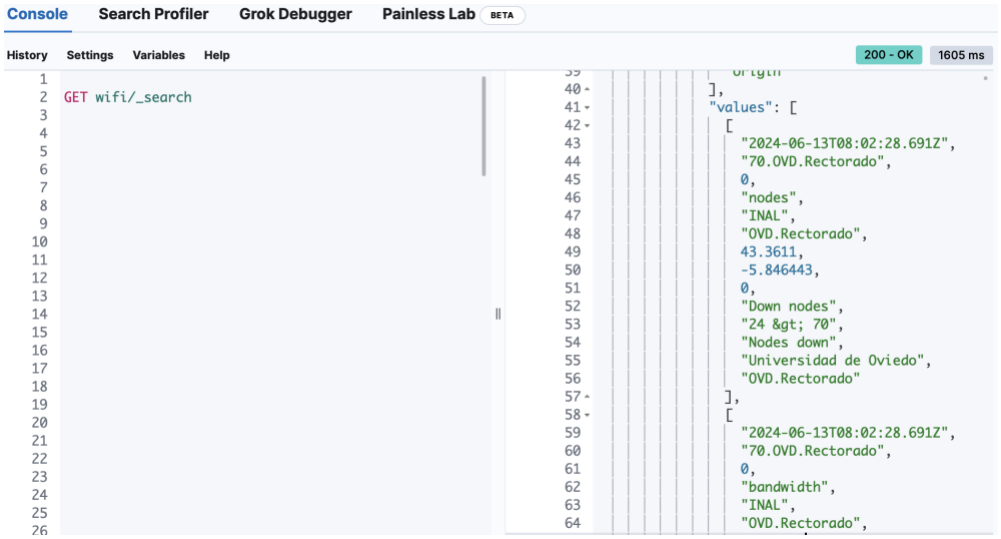


Ilustración 26. Prueba de integración wifi - Exploración

En la Ilustración 27 se observa la muestra de los datos probados juntos con todos los procesados en ese período de tiempo.

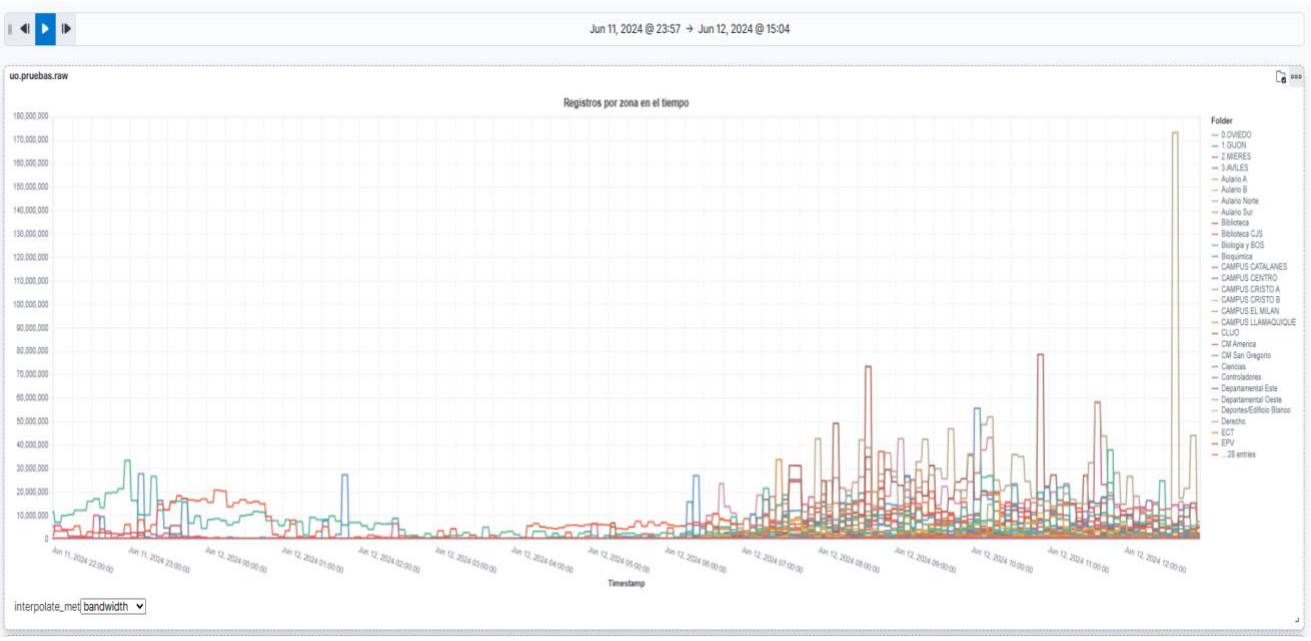


Ilustración 27 kibana uo.pruebas.raw

- **Valores Obtenidos de Dispositivos Inalámbricos LoRa (uo.pruebasdev.raw):**

En la Ilustración 28 se muestra el flujo implementado para esta variable. Hay que destacar que daremos seguimiento al Flowfile con identificador 1719246623023-55481 (Ilustración 29).

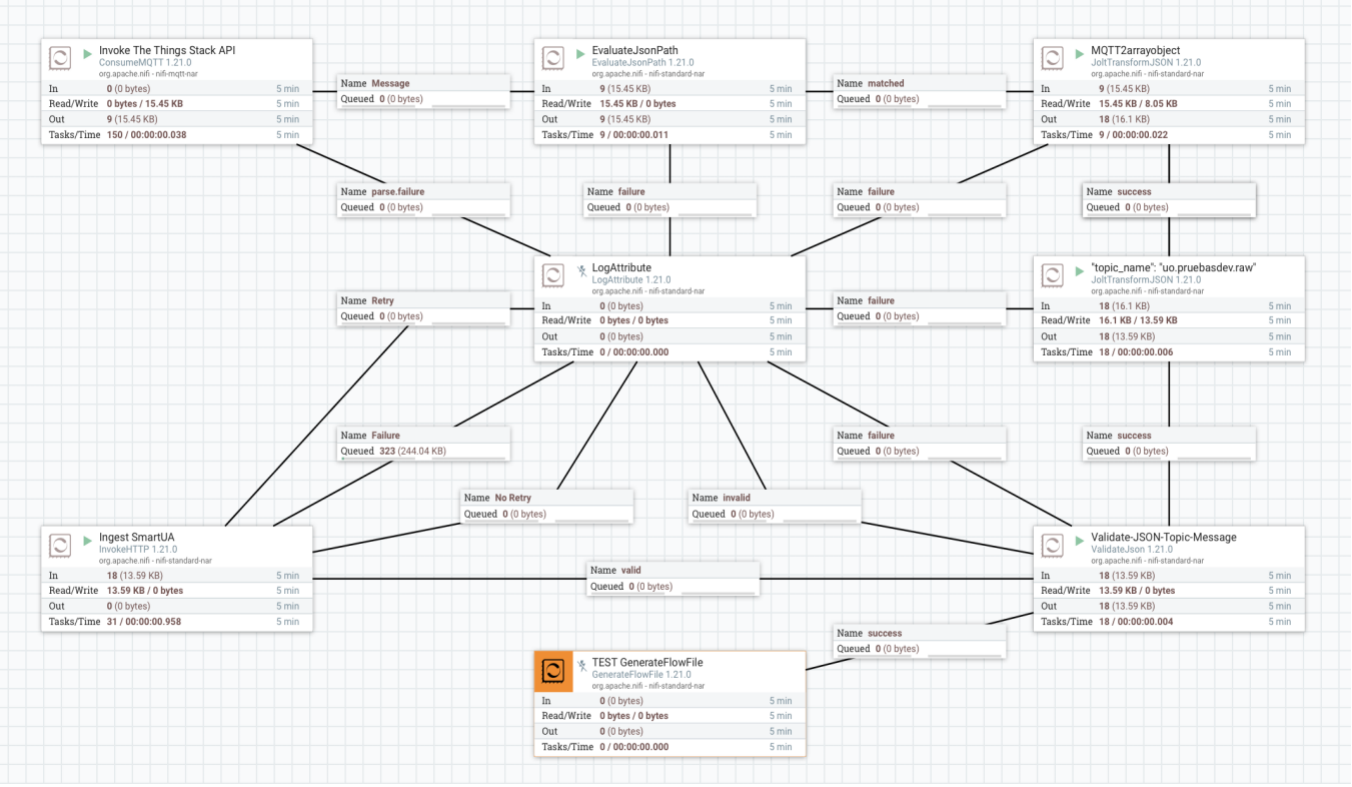


Ilustración 28. Flujo NiFi uo.pruebasdev.raw

Ilustración 29 Flowfile de seguimiento

En el anexo “Captura de datos” se adjunta la primera recepción de los datos, donde se observa los datos tal cual viene de su origen y debe ser transformado para cumplir con el esquema anteriormente

mostrado en “Esquema de datos”. Después de la captura en el anexo “Transformación” se puede ver detalladamente paso a paso el proceso por el que pasa este conjunto de datos.

Finalmente, en la Ilustración 30 se muestra cómo no llegan los datos a su destino final, en este caso no se obtuvieron los resultados esperados. En esta imagen se puede observar que en el índice definido para almacenar los datos de la variable de prueba se encuentra vacío. Lo que significa que no se han integrado los datos correctamente y por tanto no está sucediendo una correcta integración de todos los componentes tecnológicos.

Ilustración 30. Prueba de integración LoRa – exploración

Comprobado que los datos fueran validados y enviados correctamente a la api de ingesta de **kunna**, con la funcionalidad de rastreo de datos de NiFi (Ilustración 31) y la comprobación en el flujo (Ilustración 32) no queda duda de que el error ha sucedido en la integración de Kunna con Elasticsearch.

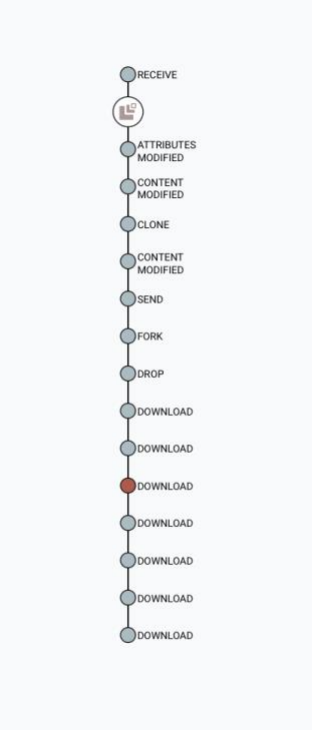


Ilustración 31. Captura tomada de Apache NiFi data provenance

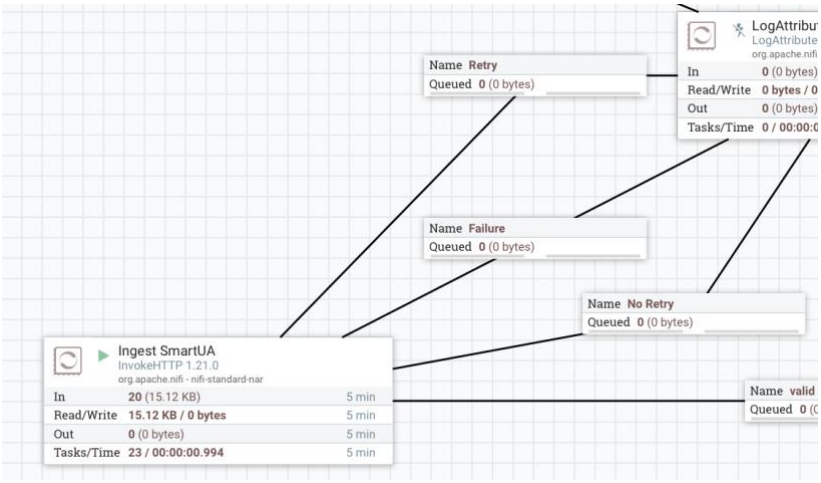


Ilustración 32. NiFi ingesta de datos a Kuna

Para solucionar este error se puso en práctica la comunicación con el grupo de desarrollo de la plataforma **kunna**, y su rápida respuesta permitió la recuperación de lo sucedido. Luego de que se nos notificara que todo debería estar funcionando cómo es debido, repetimos la última fase de la prueba y efectivamente pudimos acceder a los datos ingestados en el almacenamiento, pero ahora con otra condición, se mezclaron los datos de las dos tuberías de conexión de las variables de prueba causando que en los índices de Elasticsearch se fusionaran las dos variables, causando que la integración no fuera coherente.

Para solucionarlo se realizaron cambios en los archivos de configuración de las tuberías (*pipeline*) de Logstash y se repitió nuevamente la última fase de la prueba, obteniéndose cómo resultado la integración satisfactoria y coherente de las dos fuentes de datos sometidas a prueba. En las siguientes imágenes se muestran captura de la consola de Elasticsearch con una muestra de los datos de cada una de las fuentes y sus respectivas visualizaciones.

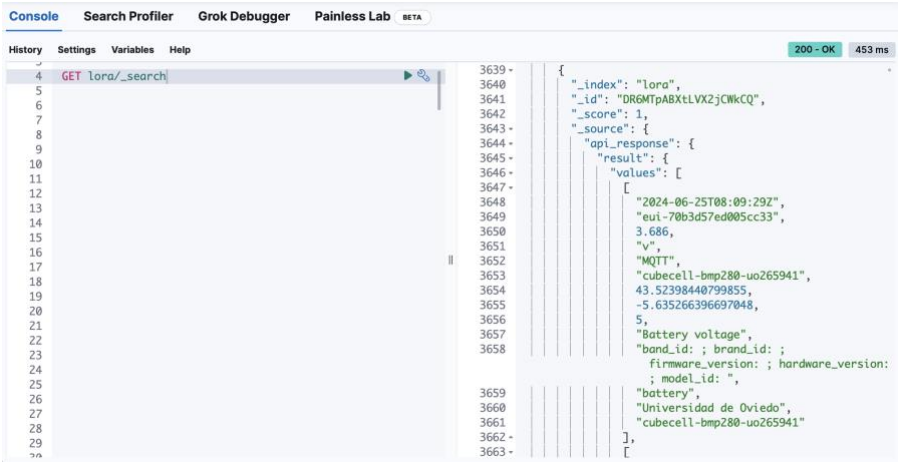


Ilustración 33. Integración fuente de dato wifi

Ilustración 34. Integración fuente de dato LoRa

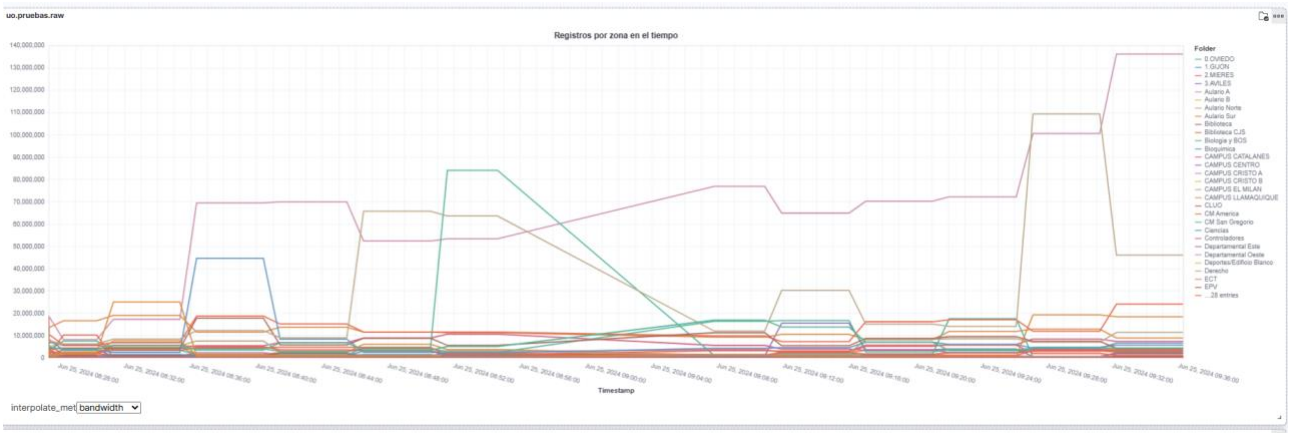


Ilustración 35. Integración fuente de dato wifi (2)

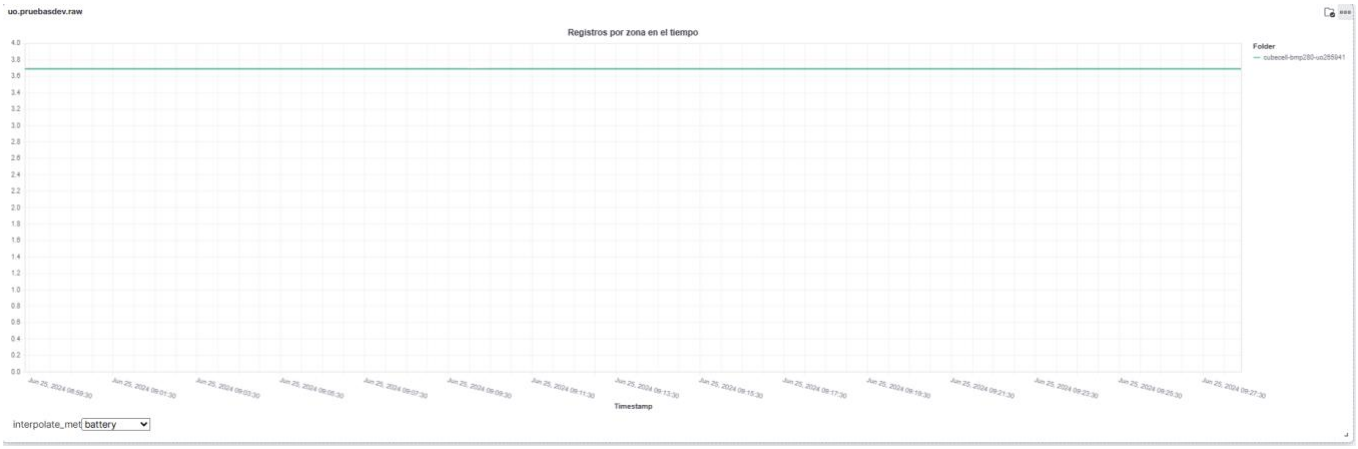


Ilustración 36. Integración fuente de dato LoRa(2)

Los resultados de esta prueba dejan clara evidencia de que los componentes del sistema se integran efectivamente.

7.3.1.2 Prueba de rendimiento

En la Ilustración 37 se muestra el flujo de NiFi. Detuvimos el *processor* InvokeHTTP (encargado de enviar los datos a kunna) para lograr acumular la mayor cantidad de FlowFiles posibles. En la imagen se observa cómo se activa la presión de retorno en casi todos los processors. Se acumularon aproximadamente 60 000 FlowFiles. En la Ilustración 39 se muestra que al pasar por el processor SplitXML (Ilustración 38) cada flowfiles se divide en 57 FlowFiles más, dando como resultado que se someterán a esta prueba aproximadamente 222 000 FlowFiles.

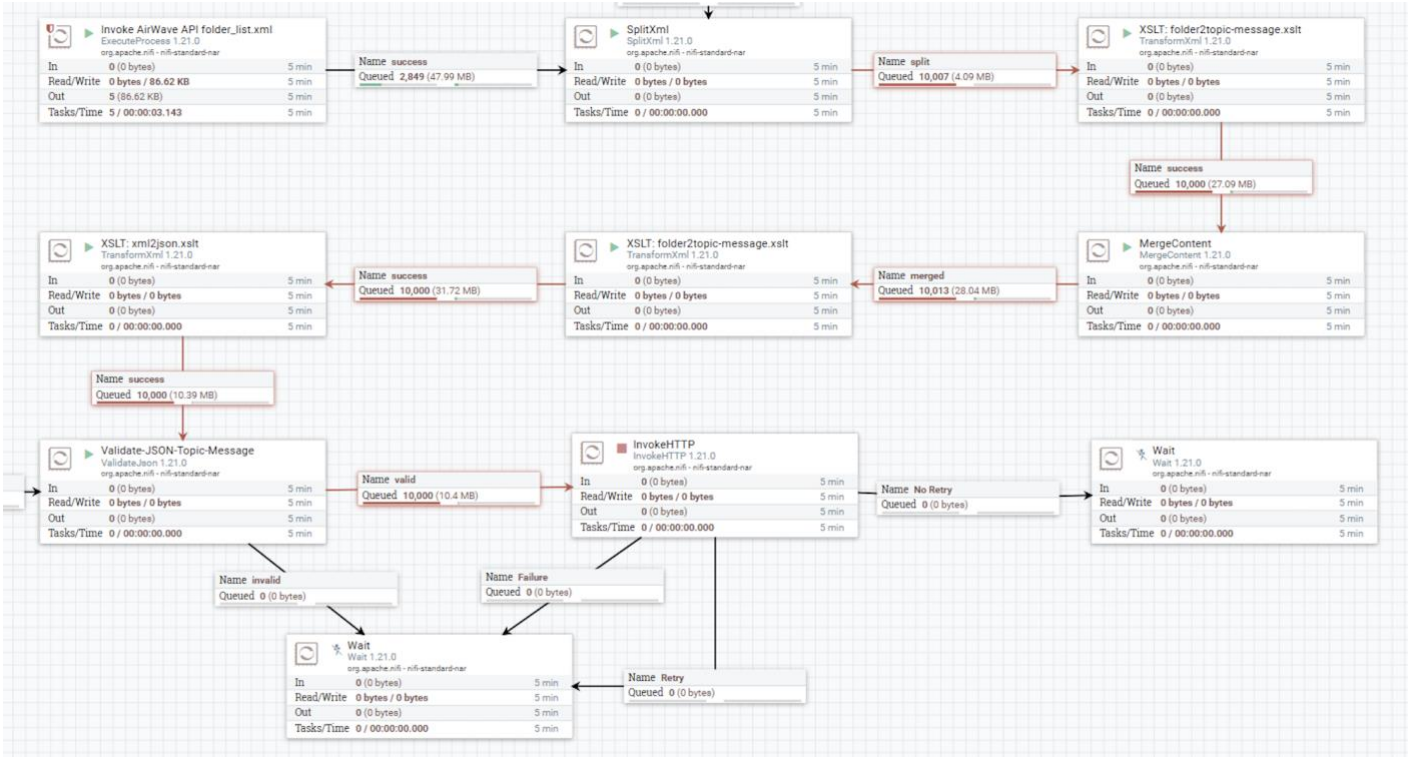


Ilustración 37.Captura del flujo de simulación en NiFi. Prueba de rendimiento

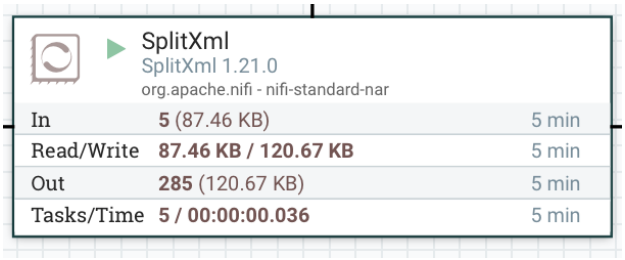


Ilustración 38. Prueba de rendimiento. SplitXML (2)

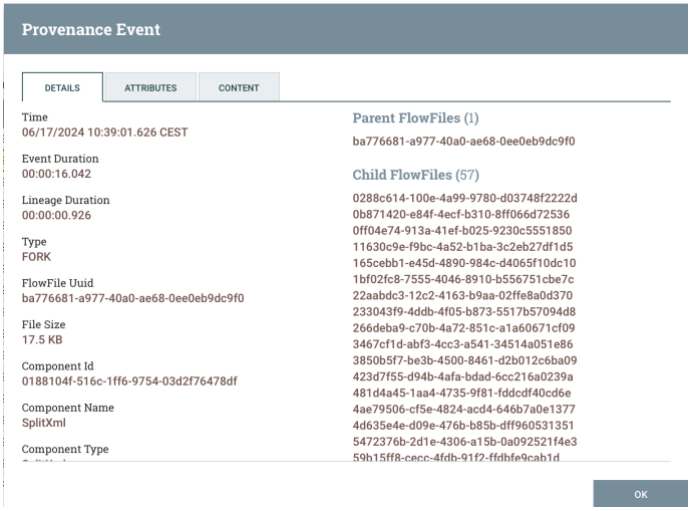


Ilustración 39. Prueba de rendimiento. SplitXML

Utilizando la funcionalidad Status History de NiFi, podemos ver detalles del procesamiento de los datos. En las Ilustración 40 e Ilustración 41 se observa cómo el processor SplitXML recibe una cantidad de FlowFiles de entrada que se multiplican a la salida según lo esperado, confirmando el correcto funcionamiento del mecanismo de división para la simulación.

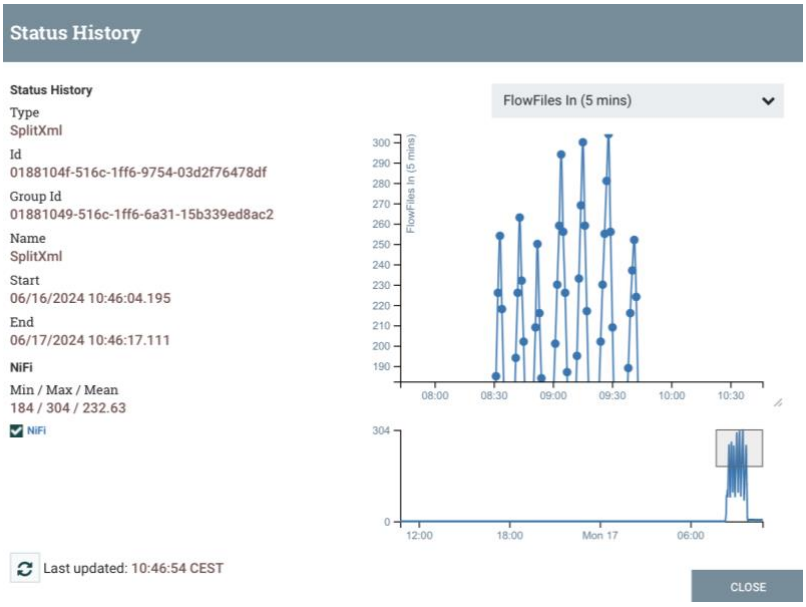


Ilustración 40. Prueba de rendimiento FlowFiles In

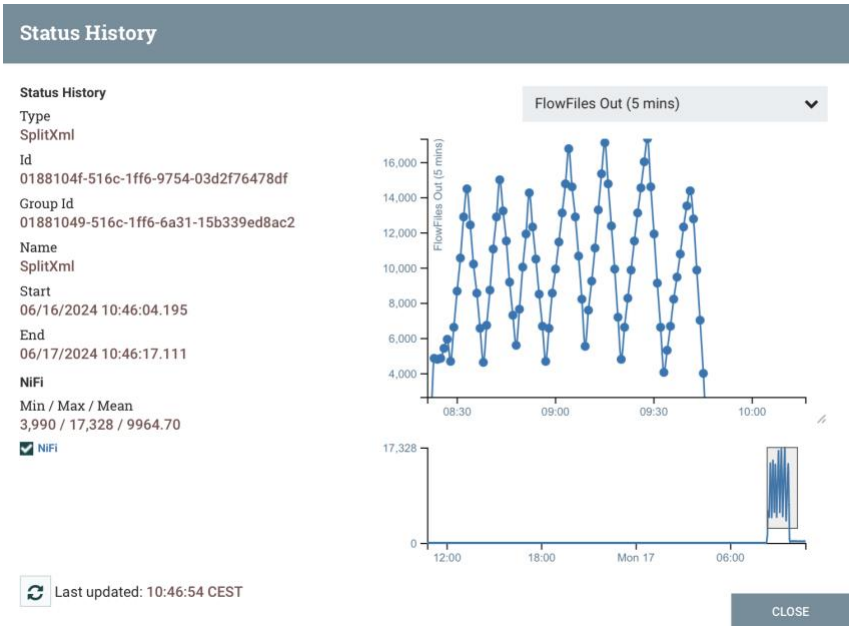


Ilustración 41. Prueba de rendimiento. FlowFiles Out

Con las condiciones de la simulación creada, iniciamos el *processor* detenido y monitoreamos la prueba. Como se observa en la Ilustración 42, el proceso inició a las 8:18 am. Este momento marca el comienzo del procesamiento masivo de FlowFiles acumulados.

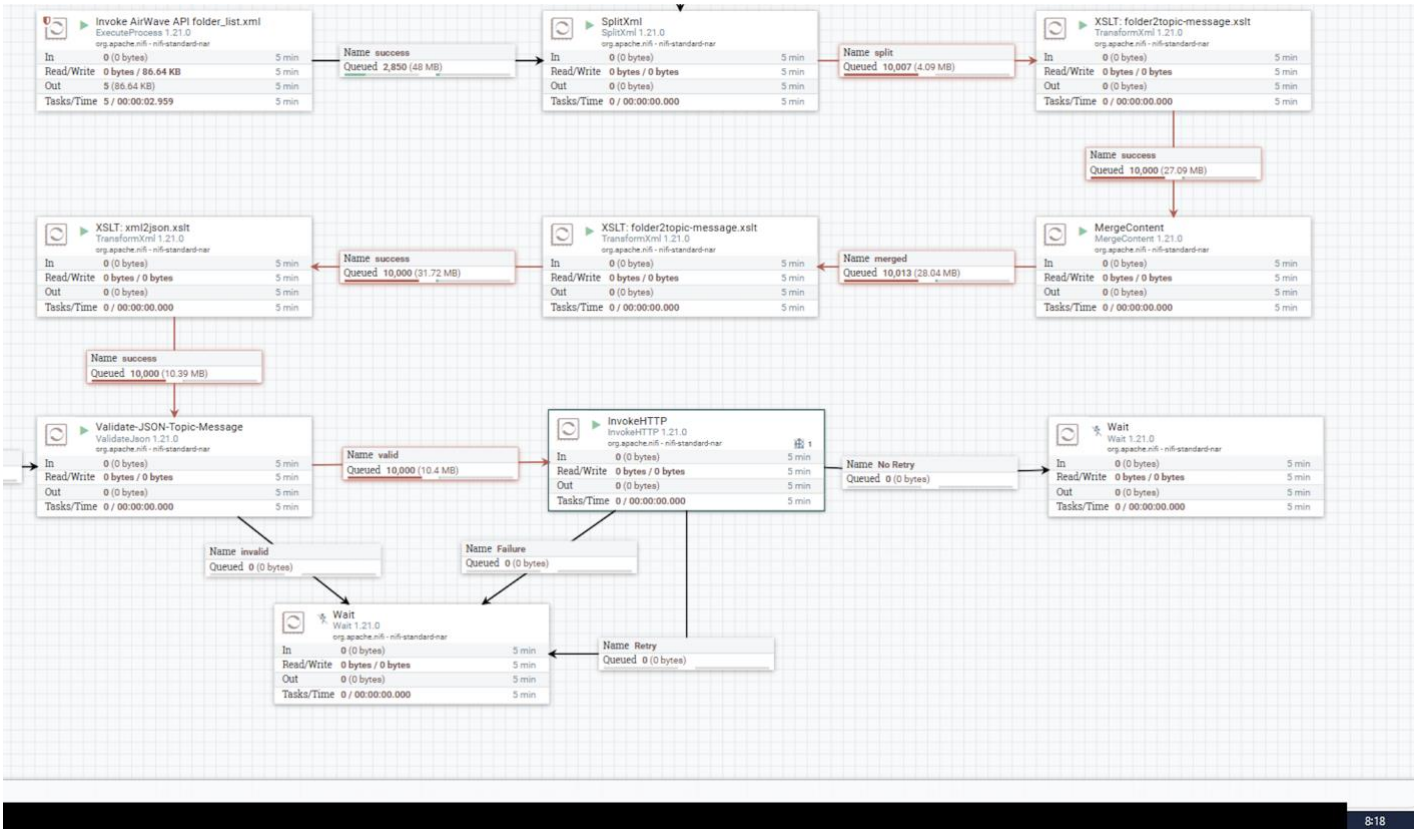


Ilustración 42. Prueba de rendimiento. Ejecución

En la Ilustración 43 al comienzo de la prueba, se observa cómo se empieza a aliviar la presión del flujo al procesar los FlowFiles encolados. Esto indica que el sistema está respondiendo adecuadamente a la carga inicial.

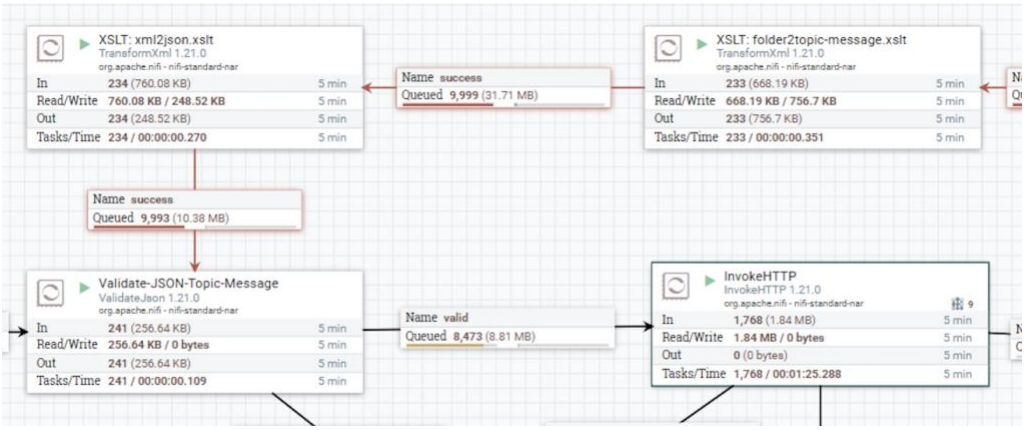


Ilustración 43. Prueba de rendimiento. Descompresión

A las 10:00 am se completó el procesamiento y envío de todos los datos. La Ilustración 44 muestra el flujo recién terminado y la hora de finalización. Este resultado demuestra la capacidad del sistema para manejar 200,000 FlowFiles en aproximadamente dos horas, sin pérdida de datos ni fallos. Esto sugiere que, en caso de pérdida de conexión con Kunna, el sistema puede recuperar un día de datos en aproximadamente dos horas.

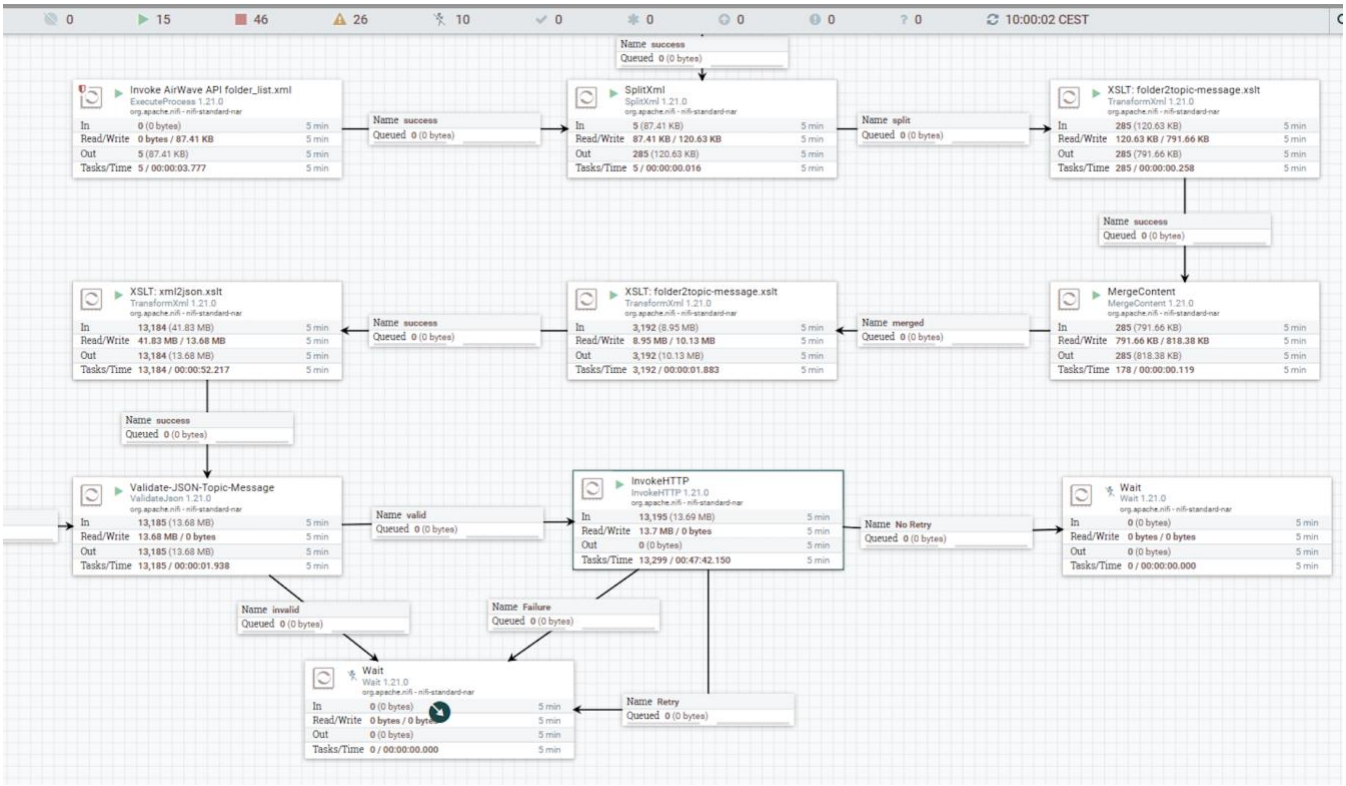


Ilustración 44. Prueba de rendimiento. Captura final de NiFi

En la Ilustración 45 se muestra el número total de subprocesos en milisegundos que el procesador ha utilizado para completar sus tareas. Si acercamos en Ilustración 46 se observa el lapso en el que NiFi logró procesar todas las tareas y cuánto tiempo le tomó. Esta información es útil para entender la eficiencia del procesador y su capacidad de manejo bajo cargas intensas.

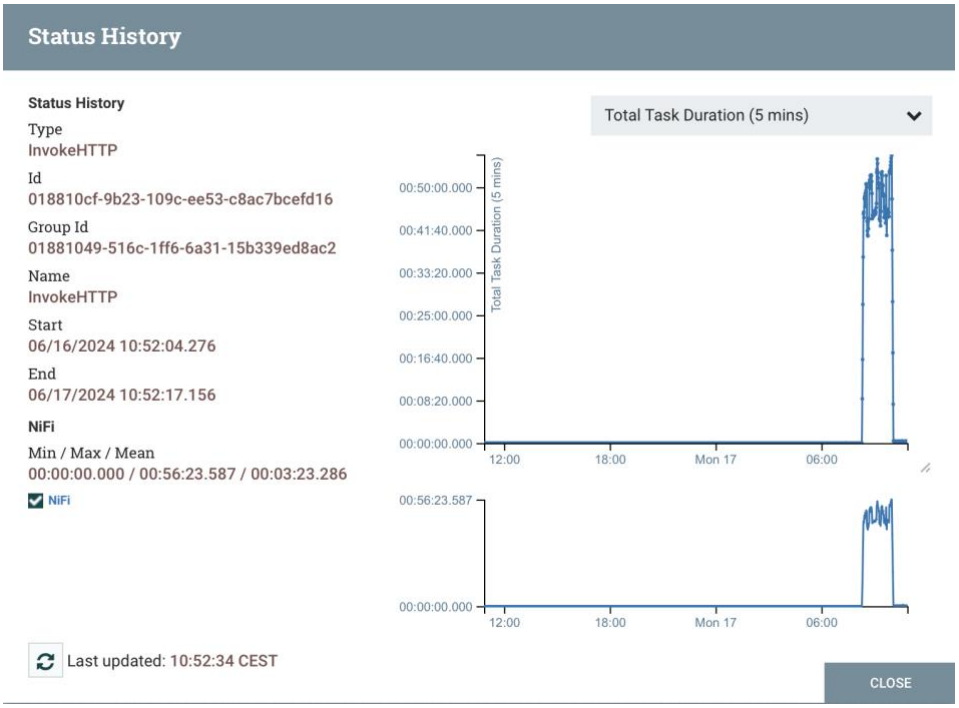


Ilustración 45. Prueba de rendimiento. Total de tareas

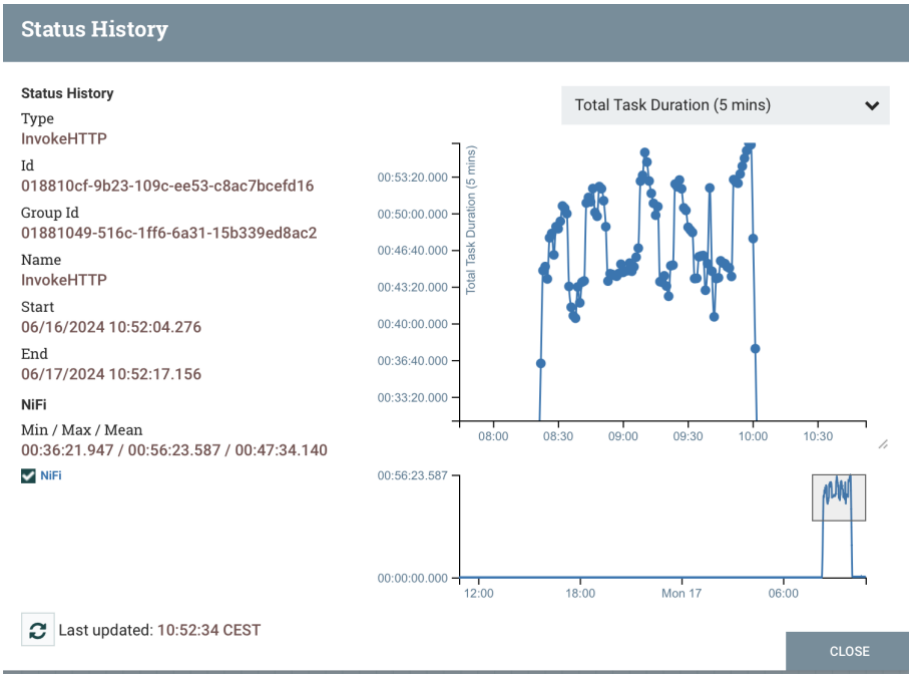


Ilustración 46. Prueba de rendimiento. Total de tareas (2)

En Ilustración 47 se muestra el número exacto de tareas completadas por el procesador InvokeHTTP durante el período de prueba. La gráfica indica que el procesador manejó entre 9,242 y 17,438 tareas en intervalos de cinco minutos, con una media de 12,963.14 tareas. Este dato nos proporciona una medida precisa de la capacidad de procesamiento del sistema, mostrando que durante los momentos

de mayor carga, el procesador fue capaz de gestionar hasta 17,438 tareas en cinco minutos, lo que equivale a una tasa de procesamiento de aproximadamente 3,487.6 tareas por minuto. Esta información es crucial para evaluar el rendimiento y la eficiencia del sistema bajo condiciones de alta carga, demostrando su capacidad para manejar grandes volúmenes de datos de manera efectiva.

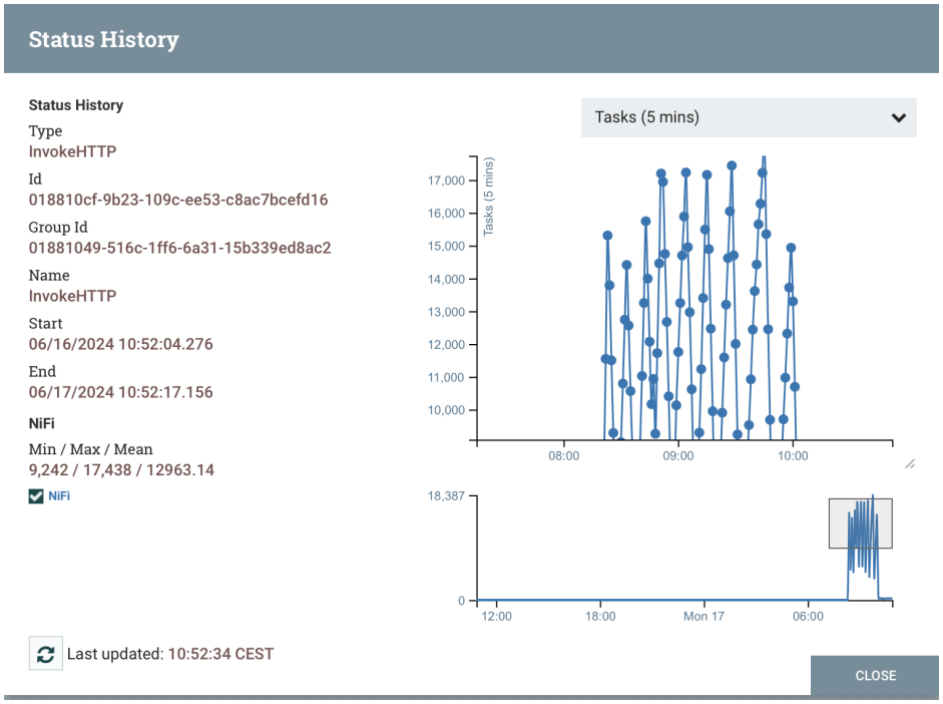


Ilustración 47. Prueba de rendimiento. Tareas

Luego de haber procesados los datos y enviados al almacenamiento, pasamos a ver cómo asimiló todos los datos la capa de visualización. Para tomar cómo punto de comparación en Ilustración 48 se muestra el estado del clúster antes de recibir los datos de la prueba. Utilizamos las herramientas de Elastic Observability para monitorear las infraestructuras, obteniendo los siguientes indicadores iniciales.

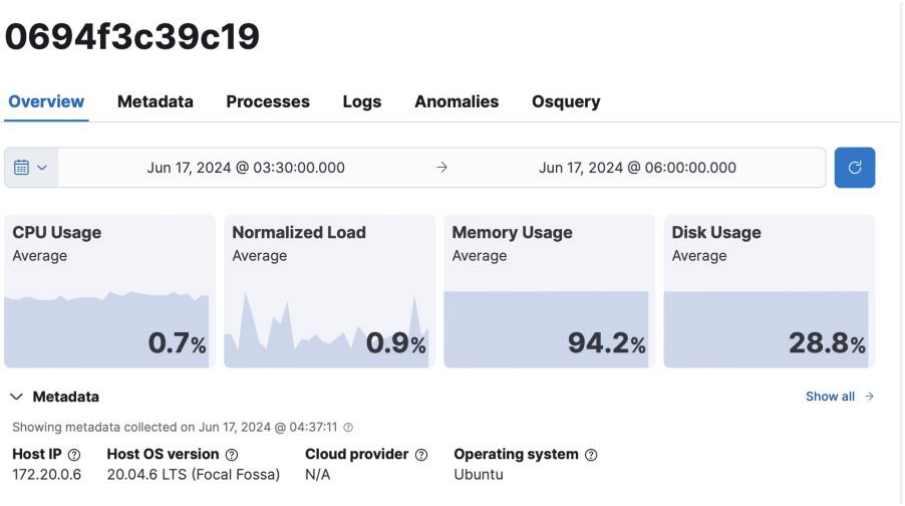


Ilustración 48. Prueba de rendimiento. Kibana

Estos indicadores proporcionan una visión clara del estado de los recursos del clúster antes de someterlo a la carga de datos de la prueba. El bajo uso de la CPU y la carga normalizada, junto con una media de memoria y disco, indican que el clúster operaba dentro de sus capacidades normales con capacidad para manejar cargas adicionales.

Después de indexar los datos de la prueba, observamos en la Ilustración 49 cómo evolucionaron estos indicadores.

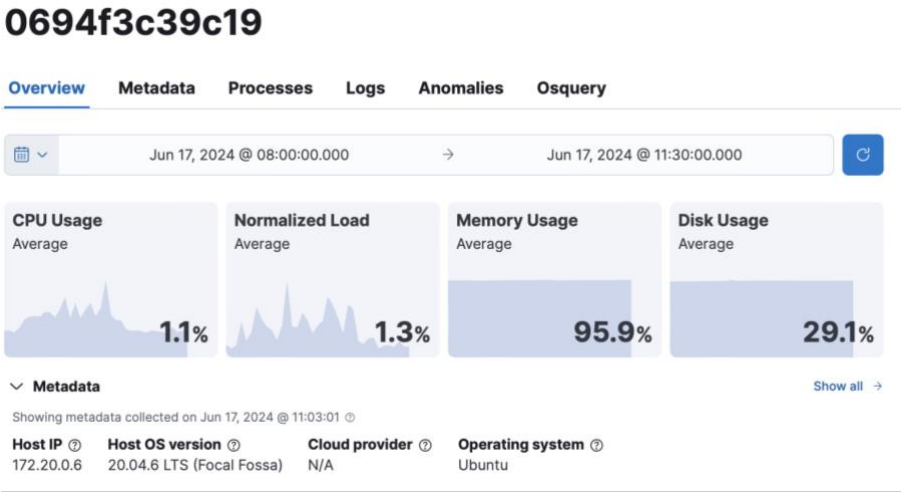


Ilustración 49. Prueba de rendimiento. Kibana (2)

Estos resultados muestran que, después de recibir y procesar los datos de la prueba, el clúster experimentó un ligero aumento en la utilización de recursos. A pesar de esto, los valores permanecen dentro de rangos aceptables, lo que indica que el sistema pudo manejar eficientemente la carga adicional sin comprometer su rendimiento ni su capacidad de respuesta.

Estos datos son críticos para evaluar la escalabilidad y robustez del sistema bajo condiciones de carga intensa. La capacidad del clúster para mantener un uso de recursos controlados y dentro de límites aceptables durante y después de la prueba demuestra su capacidad para escalar verticalmente y gestionar grandes volúmenes de datos. En el anexo “Prueba de rendimiento” se detallan las gráficas individuales correspondientes a cada valor de la imagen para una mejor comprensión.

Luego de varios días de integración de datos, se implementó la política de retención de eliminar el contenido de los índices cada 30 días, por cuestiones de almacenamiento.

La prueba de rendimiento demostró que el sistema es altamente capaz de manejar grandes volúmenes de datos, procesando exitosamente aproximadamente 200,000 FlowFiles en un período de dos horas. Los resultados indicaron que el procesador InvokeHTTP puede gestionar hasta 17,438 tareas en intervalos de cinco minutos, con una media de 12,963.14 tareas, lo que evidencia un alto rendimiento incluso bajo condiciones de alta carga. La activación y manejo efectivo de la presión de retorno aseguró la estabilidad del sistema, permitiendo el procesamiento eficiente de los datos acumulados. Además, el sistema mostró resiliencia y capacidad de recuperación rápida, procesando un día completo de datos en aproximadamente dos horas en caso de interrupciones. La observación de los recursos del clúster, utilizando Elastic Observability, indicó un ligero aumento en la utilización de CPU y memoria, pero dentro de rangos aceptables, lo que subraya la capacidad del clúster para manejar cargas adicionales sin comprometer su rendimiento.

Capítulo 8 CONCLUSIONES Y AMPLIACIONES

8.1 CONCLUSIONES

Las conclusiones derivadas del desarrollo e implantación del sistema de integración de datos institucionales heterogéneos en una plataforma abierta y distribuida son fundamentales para comprender el impacto y los resultados obtenidos en este proyecto. A continuación, se presentan las principales conclusiones:

1. **Éxitos:**
    - Se definió una arquitectura que satisface los requisitos del proyecto.
    - Se realizó un estudio de alternativas y se seleccionaron las tecnologías que mejor se adaptaban a las necesidades.
    - Se logró implementar con éxito la integración de datos provenientes de dos fuentes institucionales, permitiendo una visión unificada y enriquecida de la información.
    - Se diseñaron, especificaron y realizaron pruebas de integración y rendmiento que validaron el funcionamiento esperado de la solución propuesta.
    - La colaboración durante el proceso de desarrollo y la capacidad de interpretar los diferentes roles requeridos por el proyecto supusieron un reto significativo.
    - La utilización de tecnologías como NiFi, ELK y Kunna demostró ser efectiva para la integración, y visualización de datos en un entorno universitario.
2. **Desafíos enfrentados:**
    - Se identificaron desafíos en la gestión de permisos de acceso, problemas de almacenamiento, comunicación interdepartamental y conocimiento insuficiente sobre las tecnologías utilizadas, los cuales requirieron soluciones rápidas y efectivas.
3. **Logros alcanzados:**
    - Se construyó la base para la integración de todos los datos institucionales.
    - Se adquirieron conocimientos más allá de los obtenidos durante el proceso del máster.
    - Se estableció una base sólida para futuras expansiones y mejoras en el sistema, brindando oportunidades para seguir innovando en el ámbito de la integración de datos en entornos educativos.

En resumen, el proyecto de integración de datos institucionales ha demostrado ser un paso significativo hacia la creación de una Universidad Inteligente, donde la tecnología y la información se utilizan de manera estratégica para mejorar la experiencia educativa y la gestión institucional.

8.2 AMPLIACIONES

Las posibles ampliaciones del proyecto abarcan varias áreas clave que pueden mejorar y expandir la funcionalidad y el impacto de la plataforma de integración de datos. A continuación, se detallan las principales propuestas de ampliación:

1. **Incorporación de Más Fuentes de Datos:**
    - Ampliar la plataforma para integrar todas las fuentes de datos posibles del entorno universitario. Esto permitirá una visión más completa y detallada de la información disponible, enriqueciendo los análisis y las capacidades de toma de decisiones.
2. **Marketing y Difusión del Proyecto:**
    - Realizar campañas de marketing para aumentar la visibilidad del proyecto dentro del entorno universitario. Esto ayudará a generar interés y fomentar la participación de todos los sectores de la universidad, promoviendo el uso y aprovechamiento de la plataforma.
3. **Capacitaciones para Usuarios e Implementadores:**
    - Organizar programas de capacitación para los futuros usuarios e implementadores de la plataforma. Estas formaciones les permitirán dominar las tecnologías empleadas, facilitando el uso eficiente de la plataforma y asegurando su sostenibilidad a largo plazo.
4. **Manual del Informático:**
    - Elaborar un manual detallado para informáticos, donde se explique paso a paso cómo agregar nuevas fuentes de datos a la plataforma y cómo realizar explotaciones de estos, Este manual servirá como una guía práctica para mantener y expandir la plataforma de manera eficiente.

ANEXOS

CÓDIGO DE TRANSFORMACIONES

En este anexo se encuentra el código de las transformaciones aplicadas a las fuentes de datos durante el proceso de estandarización en Apache NiFi. Para la fuente de datos proveniente de puntos de acceso Wifi, se utilizó XSLT debido a su formato de origen en XML. En el caso de la fuente de datos proveniente de dispositivos inalámbricos LoRa a través de la red LoRaWAN, se empleó Jolt debido a su formato de origen en JSON. Estos códigos fueron implementados en el cuerpo de procesadores de NiFi para llevar a cabo las transformaciones necesarias.

Jolt

```
[
    {
        "operation": "shift",
        "spec": {
            "end_device_ids": {
                "application_ids": {
                    "application_id": [
                        "alias",
                        "origin"
                    ]
                },
                "device_id": "uid"
            },
            "uplink_message": {
                "decoded_payload": {
                    "*": {
                        "#Data Description": "data[#2].description",
                        "#Data Metric": "data[#2].metric",
                        "$": "data[#2].name",
                        "@": "data[#2].value"
                    },
```

```
"battery": {
                        "#Battery voltage": "data[#2].description",
                        "#v": "data[#2].metric",
                        "$": "data[#2].name",
                        "@": "data[#2].value"
                    },
                    "humidity": {
                        "#%": "data[#2].metric",
                        "#Humidity percentage": "data[#2].description",
                        "$": "data[#2].name",
                        "@": "data[#2].value"
                    },
"pressure": {
                        "#hPa": "data[#2].metric",
                        "#Pressure": "data[#2].description",
                        "$": "data[#2].name",
                        "@": "data[#2].value"
                    },
                    "temperature": {
                        "#°C": "data[#2].metric",
                        "#Temperature": "data[#2].description",
                        "$": "data[#2].name",
                        "@": "data[#2].value"
                    }
                },
                "locations": {
                    "user": {
                        "altitude": "cota",
                        "latitude": "lat",
                        "longitude": "lon"
                    }
                }
            },
```

```
            "version_ids": "description_origin"
        }
    },
    {
        "operation": "default",
        "spec": {
            "organizationid": "Universidad de Oviedo",
            "typemeter": "MQTT",
            "timestamp": 0,
            "timestamp_to": 0,
            "lat": 0,
            "lon": 0,
            "cota": 0,
            "timezone": "Europe/Madrid",
            "description_origin": "",
            "origin": "",
            "uid": "",
            "alias": "",
            "data": []
        }
    },
    {
        "operation": "modify-overwrite-beta",
        "spec": {
            "timestamp":
"=toNumber(${uplink_message.received_at:formatInstant('yyyy-MM-dd
HH:mm:ss.nX','GMT'):toDate('yyyy-MM-dd HH:mm:ss','GMT'):toNumber()})",
            "timestamp_to":
"=toNumber(${uplink_message.received_at:formatInstant('yyyy-MM-dd
HH:mm:ss.nX','GMT'):toDate('yyyy-MM-dd HH:mm:ss','GMT'):toNumber()})",
            "description_origin": "=concat('band_id:
',@(1,description_origin.band_id),'; brand_id:
',@(1,description_origin.brand_id),'; firmware_version:
',@(1,description_origin.firmware_version),'; hardware_version:
',@(1,description_origin.hardware_version),'; model_id:
',@(1,description_origin.model_id))"
        }
    }
]
```

```
[
  {
    "operation": "shift",
    "spec": {
      "#uo.pruebasdev.raw": "topic_name",
      "@": "arrayobjects[]"
    }
  }
]
```

XSLT

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
          xmlns:fn="http://www.w3.org/2005/xpath-functions"
          xmlns:json="http://www.w3.org/2005/xpath-functions"
          xmlns:xs="http://www.w3.org/2001/XMLSchema"
          xmlns:math="http://www.w3.org/2005/xpath-functions/math"
          xmlns:xd="http://www.oxygenxml.com/ns/doc/xsl"
          xmlns:emp="http://www.semanticalllc.com/ns/employees#"
          xmlns:h="http://www.w3.org/1999/xhtml"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xmlns:amp="http://www.airwave.com"
          exclude-result-prefixes="json fn xs math xd emp h xsi amp"
          version="3.0">

  <xsl:output method="xml" media-type="text/json" indent="yes" omit-xml-
declaration="yes"/>

  <xsl:variable name="iso-current-dateTime"
          select="xs:dateTime(format-dateTime(current-dateTime(),'[Y0001]-[M01]-
[D01]T[H01]:[m01]:[s01].[f05][Z]'))">
  </xsl:variable>
  <xsl:variable name="unix-current-dateTime"
          select="floor((xs:dateTime(current-dateTime()) - xs:dateTime('1970-01-
01T00:00:00.00+00:00')) div xs:dayTimeDuration('PT0.001S'))">
  </xsl:variable>

  <xsl:param name="topic_name"><xsl:text>uo.pruebas.raw</xsl:text></xsl:param>
  <xsl:param name="organizationid"><xsl:text>Universidad de
Oviedo</xsl:text></xsl:param>
  <xsl:param name="typemeter"><xsl:text>INAL</xsl:text></xsl:param>
  <xsl:param name="timestamp"><xsl:value-of select="$unix-current-
dateTime"/></xsl:param>
  <xsl:param name="timestamp_to"><xsl:value-of select="$unix-current-
dateTime"/></xsl:param>
  <xsl:param name="lat"><xsl:text>43.3611</xsl:text></xsl:param>
  <xsl:param name="lon"><xsl:text>-5.846443</xsl:text></xsl:param>
  <xsl:param name="cota"><xsl:text>0</xsl:text></xsl:param>
  <xsl:param name="timezone"><xsl:text>Europe/Madrid</xsl:text></xsl:param>

  <!--
 disable-output-escaping="no"
  <xsl:template match="/">
    <xsl:apply-templates/>
    <xsl:variable name="xml-output"><xsl:apply-templates/></xsl:variable>
    <xsl:copy-of select="xml-to-json($xml-output)"/>
  </xsl:template>
  -->

  <xsl:template match="/amp:amp_folder_list">
    <json:map>
      <json:string key="topic_name"><xsl:value-of
select="$topic_name"/></json:string>
      <json:array key="arrayobjects"><xsl:apply-templates
select="./folder"/></json:array>
    </json:map>
  </xsl:template>

  <xsl:template match="/json:map">
    <json:map>
      <json:string key="topic_name"><xsl:value-of
select="$topic_name"/></json:string>
      <json:array key="arrayobjects"><xsl:copy-of select="."/></json:array>
    </json:map>
  </xsl:template>
```

```
  <xsl:template match="/json:array">
    <json:map>
      <json:string key="topic_name"><xsl:value-of
select="$topic_name"/></json:string>
      <xsl:copy-of select="."/>
    </json:map>
  </xsl:template>

  <xsl:template match="folder">
    <json:map>
      <json:string key="organizationid"><xsl:value-of
select="$organizationid"/></json:string>
      <json:string key="typemeter"><xsl:value-of
select="$typemeter"/></json:string>
      <json:number key="timestamp"><xsl:value-of
select="$timestamp"/></json:number>
      <json:number key="timestamp_to"><xsl:value-of
select="$timestamp_to"/></json:number>
      <json:number key="lat"><xsl:value-of select="$lat"/></json:number>
      <json:number key="lon"><xsl:value-of select="$lon"/></json:number>
      <json:number key="cota"><xsl:value-of select="$cota"/></json:number>
      <json:string key="timezone"><xsl:value-of
select="$timezone"/></json:string>
      <json:string key="description_origin">
     <xsl:value-of select="./parent_id"/>
     <xsl:text> &gt; </xsl:text>
     <xsl:value-of select="./@id"/>
      </json:string>
      <json:string key="origin"><xsl:value-of select="./name"/></json:string>
      <json:string key="uid">
     <xsl:value-of select="./@id"/>
     <xsl:text>.</xsl:text>
     <xsl:value-of select="./name"/>
      </json:string>
      <json:string key="alias"><xsl:value-of select="./name"/></json:string>
      <json:array key="data">
     <xsl:apply-templates select="./client_count"/>
     <xsl:apply-templates select="./vpn_client_count"/>
     <xsl:apply-templates select="./bandwidth_in"/>
     <xsl:apply-templates select="./bandwidth_out"/>
     <xsl:apply-templates select="./up"/>
     <xsl:apply-templates select="./down"/>
     <xsl:apply-templates select="./mismatch"/>
      </json:array>
    </json:map>
  </xsl:template>

  <xsl:template match="bandwidth_in">
    <json:map>
      <json:string key="name">Node bandwidth_in</json:string>
      <json:number key="value"><xsl:value-of select="."/></json:number>
      <json:string key="metric">bandwidth</json:string>
```

```
<xsl:template match="bandwidth_out">
    <json:map>
      <json:string key="name">Node bandwidth_out</json:string>
      <json:number key="value"><xsl:value-of select="."/></json:number>
      <json:string key="metric">bandwidth</json:string>
      <json:string key="description">Out bandwidth</json:string>
    </json:map>
  </xsl:template>

  <xsl:template match="client_count">
    <json:map>
      <json:string key="name">Node connections</json:string>
      <json:number key="value"><xsl:value-of select="."/></json:number>
      <json:string key="metric">connections</json:string>
      <json:string key="description">Total number of users associated to the
device regardless of which radio they are associated to, at the time of the
last polling.</json:string>
    </json:map>
  </xsl:template>

  <xsl:template match="down">
    <json:map>
      <json:string key="name">Nodes down</json:string>
      <json:number key="value"><xsl:value-of select="."/></json:number>
      <json:string key="metric">nodes</json:string>
      <json:string key="description">Down nodes</json:string>
    </json:map>
  </xsl:template>

  <xsl:template match="mismatch">
    <json:map>
      <json:string key="name">Nodes mismatch</json:string>
      <json:number key="value"><xsl:value-of select="."/></json:number>
      <json:string key="metric">nodes</json:string>
      <json:string key="description">Mismatch nodes</json:string>
    </json:map>
  </xsl:template>

  <xsl:template match="up">
    <json:map>
      <json:string key="name">Nodes up</json:string>
      <json:number key="value"><xsl:value-of select="."/></json:number>
      <json:string key="metric">nodes</json:string>
      <json:string key="description">Up nodes</json:string>
    </json:map>
  </xsl:template>
<xsl:template match="vpn_client_count">
    <json:map>
      <json:string key="name">Node vpn_connections</json:string>
      <json:number key="value"><xsl:value-of select="."/></json:number>
      <json:string key="metric">connections</json:string>
      <json:string key="description">Number of VPN connections</json:string>
    </json:map>
  </xsl:template>

</xsl:stylesheet>
```

En el siguiente anexo (xml2json.xslt) se muestra el XSLT diseñado para convertir un documento XML en formato JSON. Utiliza varias funciones y espacios de nombres para realizar esta

transformación. La plantilla principal selecciona todo el contenido del documento XML y lo convierte a JSON con la función `xml-to-json`.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
          xmlns:xs="http://www.w3.org/2001/XMLSchema"
          xmlns:math="http://www.w3.org/2005/xpath-functions/math"
          xmlns:xd="http://www.oxygenxml.com/ns/doc/xsl"
          xmlns:emp="http://www.semanticalllc.com/ns/employees#"
          xmlns:h="http://www.w3.org/1999/xhtml"
          xmlns:fn="http://www.w3.org/2005/xpath-functions"
          xmlns:j="http://www.w3.org/2005/xpath-functions"
          exclude-result-prefixes="xs math xd h emp"
          version="3.0">
  <xsl:output method="xml" media-type="text/json" indent="yes" omit-
xml-declaration="yes"/>

  <xsl:template match="/">
    <xsl:copy-of select="xml-to-json(.)"/>
  </xsl:template>

</xsl:stylesheet>
```

CÓDIGO DE VISUALIZACIONES AVANZADAS

Vega-Lite

El siguiente anexo adjunta el código en Vega-Lite de unas de las visualizaciones implementadas en Kibana .

```
{
  "$schema": "https://vega.github.io/schema/vega-lite/v4.json",
  "config": {
    "kibana": {
      "tooltips": {
        "position": "top",
        "padding": 15
      }
    }
  },
  "title": "Registros por zona en el tiempo",
  "data": {
    "name": "source",
    "url": {
      "index": "wifi",
      "body": {
        "size": 1000,
        "query": {
          "bool": {
            "must": [
              "%dashboard_context-must_clause%",
              {
                "range": {
                  "@timestamp": {
                    "%timefilter%": true,

                  }
                }
              }
            ],
            "must_not": [
              "%dashboard_context-must_not_clause%"
            ],
            "filter": [
              "%dashboard_context-filter_clause%"
            ]
          }
```

```
        },
"_source": [
          "api_response.result.values"
        ]
      }
    },
    "format": {
      "property": "hits.hits"
    }
  },
   "params": [
    {
      "name": "interpolate",
      "select": {"type": "point", "fields": ["metric"]},
      "value":"bandwidth",
      "bind": {"input": "select", "options": ["nodes", "connections", "bandwidth"]}
    },{
    "name": "grid",
    "select": "interval",
    "bind": "scales"
  }



  ],
  "transform": [
    {
      "flatten": ["_source.api_response.result.values"],
      "as": ["flattened_values"]
    },
    {
      calculate: toDate(datum.flattened_values[0])
      as: timestamp
    },
    {
      "calculate": "datum.flattened_values[1]",
      as: "uid"
    },
    {
      "calculate": "datum.flattened_values[4]",
      as: "typemeter"
    },
```

```
{
  "calculate": "datum.flattened_values[5]",
      as: "alias"
    },
    {
    "calculate": "datum.flattened_values[6]",
      as: "lat"
    },
    {
      "calculate": "datum.flattened_values[7]",
      as: "lon"
    },
    {
      "calculate": "datum.flattened_values[8]",
      as: "cota"
    },
    {
    "calculate": "datum.flattened_values[9]",
      as: "description"
    },
    {
  "calculate": "datum.flattened_values[10]"
      as: "description_origin",
    }
    {
      "calculate": "datum.flattened_values[11]",
      as: "name"
    }
    {
     "calculate": "datum.flattened_values[12]",
      as: "organizationid"
    }
    {
      "calculate": "datum.flattened_values[13]",
      "as": "origin"
    },
     {
      "calculate": "datum.flattened_values[2]",
      "as": "value"
    },
    {
```

```
      "calculate": "datum.flattened_values[3]",
      "as": "metric"
    },
    {
    "timeUnit": "utcyearmonthdatehoursminutesseconds",
    "field": "timestamp",
    "as": "time"
  },



     {

       "filter": {
          "and": [{"param": "interpolate", "equals": "datum.metric"},
          ]

    },
    }
  {
      "joinaggregate": [

        {"op": "average", "field": "value", "as": "averageValue"}
      ],
      "groupby": ["time", "origin"]
    }
  ],
  "mark": "line",
  "encoding": {
    "x": {
      "field": "time",
      "timeUnit": "utcyearmonthdatehoursminutesseconds",
      "title": "Timestamp",
      "axis": { "labelAngle": 15 }
    },
    "y": {
      "field": "averageValue",
      "type": "quantitative",
      "title": ""
    },
    "color": {
```

```
"field": "origin",
      "type": "nominal",
      "title": "Folder",

    },"tooltip": [
    {"field": "timestamp", "type": "temporal", "title": "Tiempo"},
      {"field": "uid", "type": "nominal", "title": "uid"},
       {"field": "metric", "type": "nominal", "title": "metric"},
           {"field": "typemeter", "type": "nominal", "title": "typemeter"},
               {"field": "alias", "type": "nominal", "title": "alias"},
                   {"field": "lat", "type": "nominal", "title": "lat"},
                       {"field": "lon", "type": "nominal", "title": "lon"},
                           {"field": "cota", "type": "nominal", "title": "cota"},
                               {"field": "description", "type": "nominal", "title":
"description"},
                                   {"field": "name", "type": "nominal", "title":
"name"},

                                       {"field": "organizationid", "type":
"nominal", "title": "organizationid"},
    {"field": "origin", "type": "nominal", "title": "origin"},
    {"field": "averageValue", "type": "quantitative", "title": "Valor Promedio"}

  ]
  }
}
```

ESQUEMA DE VALIDACIÓN ESTÁNDAR

```
{
  "$id": "https://backoffice.smartua.es/ingest-data.schema.json",
  "$schema": "https://json-schema.org/draft/2020-12/schema",
  "title": "ingest-data",
  "description": "Ingest data schema for SmartUA plataform",
  "type": "object",
  "properties": {
    "topic_name": {
      "type": "string",
      "description": "Name of topic which you can ingest data"
    },
    "arrayobjects": {
      "type": "array",
      "items": [
        {
          "type": "object",
          "properties": {
            "organizationid": {
              "type": "string",
              "description": "Name of your organization"
            },
            "typemeter": {
              "type": "string",
              "description": "Name of type of meter"
            },
            "timestamp": {
              "type": "integer",
              "description": "Initial time in milliseconds (required in ms)"
            },
            "timestamp_to": {
              "type": "integer",
              "description": "Final time in milliseconds (required in ms)"
            },
            "lat": {
              "type": "number",
              "description": "Latitude (double)"
            },
            "lon": {
              "type": "number",
              "description": "Longitude (double)"
            },
            "cota": {
              "type": "integer",
              "description": "Altitude (double)"
            },
            "timezone": {
              "type": "string",
              "description": "Your time zone"
            },
```

```
"description_origin": {
              "type": "string",
              "description": "The description of this data"
            },
            "origin": {
              "type": "string",
              "description": "Origin's name of data"
            },
            "uid": {
              "type": "string",
              "description": "UID unique name"
            },
            "alias": {
              "type": "string",
              "description": "Alias"
            },
            "data": {
              "type": "array",
              "items": [
                {
                  "type": "object",
                  "properties": {
                    "name": {
                      "type": "string",
                      "description": "Name of data"
                    },
                    "value": {
                      "type": "number",
                      "description": "Value of data (double)"
                    },
                    "metric": {
                      "type": "string",
                      "description": "Metric of data"
                    },
                    "description": {
                      "type": "string",
                      "description": "Description of this value"
                    }
                  },
                  "required": [
                    "name",
                    "value",
                    "metric",
                    "description"
                  ]
                }
              ]
            }
          },
          "required": [
            "organizationid",
            "typemeter",
            "timestamp",
            "timestamp_to",
            "lat",
            "lon",
            "cota",
            "timezone",
            "description_origin",
            "origin",
            "uid",
```

```
"alias",
            "data"
          ]
        }
      ]
    }
  },
  "required": [
    "topic_name",
    "arrayobjects"
  ]
}
```

PLAN DE GESTIÓN DE RIESGOS

En este anexo se adjunta el plan de gestión de riesgos realizada para este proyecto. En el plan adjunto se incluye una descripción de la metodología seguida, las herramientas y las tecnologías, los roles, el presupuesto, un desglose de las categorías de riesgo, las definiciones de probabilidad, la matriz de probabilidad de impacto empleada, los planes de contingencia, el formato de la documentación

Título del proyecto:	***Integración de datos institucionales heterogéneos en plataforma abierta y distribuida***	**Fecha:**	15/01/2024

Cambios

Cambios en la versión 2.0

Lista de cambios.

METODOLOGÍA:

Este Plan de Gestión de Riesgos establece la estrategia para abordar los riesgos identificados en el proyecto. La identificación de riesgos es una tarea continua que involucra a todos los socios a lo largo del desarrollo del proyecto, ya que los riesgos son dinámicos y evolucionan con el tiempo.

La metodología para la gestión de riesgos comprende dos aspectos distintos:

Metodología General. Metodología para el conjunto del Plan de Gestión de Riesgos.

- **Metodología de Gestión de Riesgos**. Metodología para el ciclo de vida de todos los riesgos y las interrelaciones entre los riesgos identificados.
- **Metodología general**

La Metodología general describe el proceso de gestión de riesgos para el conjunto del proyecto: desde el inicio hasta el final de los trabajos relacionados con el proyecto.

1) **Identificación Inicial de Riesgos**: Se realiza un primer acercamiento a los riesgos del proyecto, utilizando sesiones de tormenta de ideas para identificar posibles amenazas y oportunidades.
2) **Elaboración del Plan de Gestión de Riesgos**: Se crea el plan, los criterios para la gestión de riesgos, el Registro de Riesgos y la Hoja de Datos de riesgos para los riesgos priorizados.
3) **Monitorización Continua de Riesgos**:
    a. Evaluación Regular de Riesgos: Se evalúan periódicamente los riesgos para garantizar que no representen una amenaza para el progreso del proyecto.

b. Actualización del Plan de Gestión de Riesgos: A medida que el proyecto avanza, se actualiza el plan para incorporar nuevos riesgos y ajustar estrategias según sea necesario.

c. Proyecto de Actualización del Plan por Decisiones de Evaluación de Riesgos: Se toman decisiones durante el proyecto para evitar o minimizar el impacto de los riesgos.

4) **Informe Final de Riesgos**: Al concluir el proyecto, se presenta un informe final que documenta la evolución de los riesgos a lo largo del ciclo del proyecto.

- Metodología de Gestión de Riesgos

Los aspectos más técnicos de la gestión de riesgos están involucrados en este aspecto de la metodología. Para ese proyecto definimos seis pasos:

1) **Planificar la Gestión de Riesgos**: Define cómo llevar a cabo las actividades de gestión de riesgos para el proyecto.
2) **Identificar los Riesgos**: Determina los riesgos que pueden afectar el proyecto y documenta sus características.
3) **Realizar el Análisis Cualitativo de Riesgos**: Prioriza los riesgos para análisis adicionales, evaluando la probabilidad de ocurrencia y el impacto.
4) **Realizar el Análisis Cuantitativo de Riesgos**: Analiza numéricamente el efecto de los riesgos en los objetivos generales del proyecto.
5) **Planificar la Respuesta a los Riesgos**: Desarrolla opciones y acciones para mejorar oportunidades y reducir amenazas a los objetivos del proyecto.
6) **Monitorizar y Controlar los Riesgos:** Implementa planes de respuesta, rastrea riesgos identificados, monitoriza riesgos residuales, identifica nuevos riesgos y evalúa la efectividad del proceso a lo largo del proyecto.

Conceptos generales

Con el fin de evitar la ambigüedad, se deben definir tres conceptos utilizados en la Gestión de Riesgos:

- **Riesgo**: es cualquier evento que pueda causar un efecto en el proyecto. El efecto bien podría ser una amenaza o podría ser una oportunidad. La primera de ellas debe ser evitada y la segunda debe ser explotada.
- **Factor de riesgo**: Los factores de riesgo son circunstancias asociadas con el riesgo de que aumenta la posibilidad de conseguir los efectos descritos por el riesgo. Por lo general, los riesgos son imposibles o difíciles de medir o controlar directamente y las decisiones se toman controlando los factores de riesgo con el fin de evitar o minimizar (o potenciar) los efectos de riesgo.
- **Indicadores de Riesgo**: Los indicadores de riesgo son elementos asociados al riesgo y / o sus factores, que se puede medir y sirve como información acerca de la posibilidad de que se produzca el efecto del riesgo.

HERRAMIENTAS Y TÉCNOLOGÍAS

Se utilizarán diferentes técnicas para la gestión de riesgos.

- Tormenta de Ideas

Esta técnica será utilizada en un principio para la identificación de la lista principal de los riesgos. Consiste en la discusión acerca de los principales objetivos del proyecto y todas las cosas que pueden ir mal.

- **Las evaluaciones periódicas**

Regularmente, todos los indicadores serán evaluados de acuerdo con el plan establecido en la hoja de riesgo. Todos estos resultados se discutirán en las reuniones de riesgos.

- **Reuniones**

Regularmente, se debe incluir un punto de discusión en el orden del día de las reuniones con el fin de tomar decisiones acerca de los riesgos del proyecto y sus efectos.

- **Delphi**

En caso de desacuerdo sobre cualquier riesgo o decisión, se puede un método Delphi utilizar para resolver el problema.

- **Diagramas causa-efecto**

Es necesaria, debido a la complejidad de las relaciones entre un riesgo y sus factores de riesgo, un diagrama de causa-efecto se puede utilizar con el fin de mostrar una representación gráfica.

- **Cálculos Estadísticos**

En muchos casos se requerirán cálculos estadísticos para resolver la evaluación cuantitativa.

- **Otros**

Se usarán varios tipos de herramientas y procesos matemáticos para hacer las evaluaciones cuantitativas.

ROLES Y RESPONSIBILIDADES

Se prevén lo siguientes roles:

1. **Responsable de riesgos**: Responsable de la gestión de riesgos. Este papel lo jugará el propio jefe de Proyecto.
2. **Auxiliar de riesgos**: Personas encargadas del seguimiento y monitorización regular de los riesgos (analista de datos.)
3. **Identificador de riesgos**: Todo el equipo es responsable de la identificación de nuevos riesgos.

PRESUPUESTO

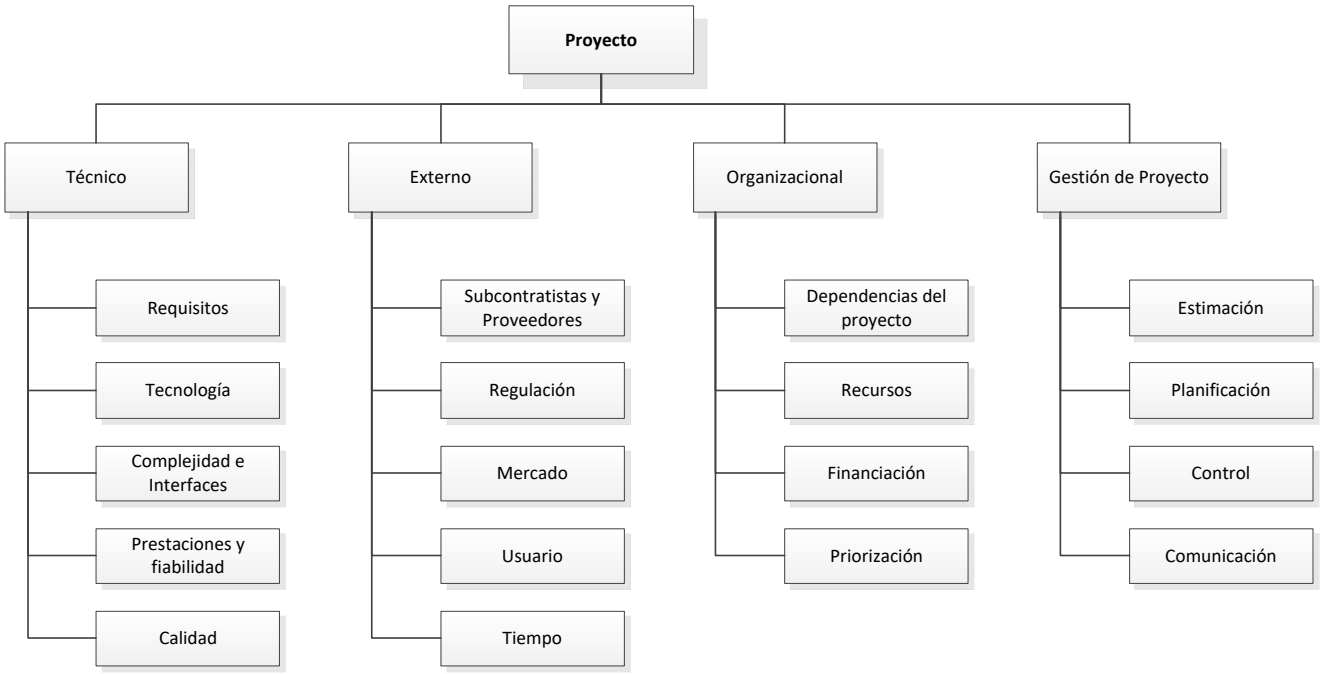
Item	Concepto	Asignación($)

1	Identification de los riesgos	150
2	Análisis y priorización de los datos	200
3	Planificación de los riegos	130
4	Definición de planes de contingencia	500
5	Actualización y monitorización de los riesgos	200
	Total	1180

CALENDARIO

Hito/Actividad	**Fecha**
Identificación de los riegos	1/11/2023
Análisis y priorización de los riegos	3/11/2023
Planificación de los riesgos	5/112023
Definición de planes de contingencia	5/11/2023

CATEGORÍAS DE RIESGO



Esta estructura de desglose de riesgo (Risk Breakdown Structure o RBS) lista las categorías y subcategorías en las que los riesgos surgen para un proyecto títpio. Dependiendo del tipo de proyecto y del tipo de organización, el RBS puede ser diferente. Una de las ventajas de este enfoque es para recordar a los participantes en un ejercicio de identificación de riesgos de las muchas fuentes de donde pueden surgir los riesgos del proyecto.

DEFINICIONES DE PROBABILIDAD

Muy Baja	(0%..20%] El valor usado en la matriz de probabilidad e impacto es: 10%
Baja	(20%..40%] El valor usado en la matriz de probabilidad e impacto es: 30%
Madia	(40%..60%] El valor usado en la matriz de probabilidad e impacto es: 50%
Alta	(60%..80%] El valor usado en la matriz de probabilidad e impacto es: 70%
Muy Alta	(80%..100%) El valor usado en la matriz de probabilidad e impacto es: 90%

DEFINICIONES DE IMPACTO POR OBJETIVOS

Condiciones definidas para las escalas de impacto de un riesgo sobre los objetivos principales del proyecto (Se muestran ejemplos para impactos negativos únicamente)					
Objetivos de proyecto	Escalas relativas o numéricas				
	Muy bajo / 5%	Bajo / 10%	Moderado / 20%	Alto / 40%	Muy alto / 80%
Coste	Incremento del coste insignificante	Incremento del coste <10%	Incremento del coste entre el 10-20%	Incremento del coste entre el 20-40%	Incremento del coste >40%
Tiempo	Incremento de tiempo insignificante	Incremento de tiempo <5%	Incremento de tiempo entre el 5-10%	Incremento de tiempo entre el 10-20%	Incremento de tiempo >20%
Alcance	Reducciones del alcance inapreciables	Afectadas áreas poco importantes del alcance	Afectadas áreas importantes del alcance	Reducciones del alcance inaceptables para el cliente	El resultado final del proyecto no es realmente útil
Calidad	La degradación de la calidad es inapreciable	Sólo las aplicaciones muy exigentes se ven afectadas	La reducción de la calidad requiere la aceptación del cliente	Reducción de la calidad inaceptable para el cliente	El resultado final del proyecto no es realmente útil
Esta tabla presenta ejemplos de definiciones impacto de riesgo para cuatro obejtivos de proyecto diferentes. Debe ser ajustado en el proceso de elaboración del Plan de Gestión de Riesgos a cada proyecto concreto y a lo sumbrales de riesgo de la organización. Las definiciones del impacto deben ser desarrolladas para los riesgos positivos (oportunidades) de una manera similar.					

MATRIZ DE PROBABILIDAD E IMPACTO

Probabilidad	**Muy Alta**	***0,90***	0,05	0,14	0,27	0,50	0,81
	Alta	***0,70***	0,04	0,11	0,21	0,39	0,63
	Media	***0,50***	0,03	0,08	0,15	0,28	0,45
	Baja	***0,30***	0,02	0,05	0,09	0,17	0,27
	Muy Baja	***0,10***	0,01	0,02	0,03	0,06	0,09
			0,05	***0,15***	***0,30***	***0,55***	***0,90***
			Muy Bajo	**Bajo**	**Medio**	**Alto**	**Crítico**
			Impacto				

NIVELES DE TOLERANCIA

La tolerancia está en 0,3

PLANES DE CONTINGENCIA

12.1 Presupuesto

En situaciones que demanden ajustes al presupuesto inicialmente definido y acordado antes del inicio del proyecto, se procederá a cuantificar y evaluar dichos ajustes en colaboración entre el director del proyecto y el jefe de proyectos. Cualquier modificación presupuestaria no deberá superar el 8% del costo total del proyecto.

12.2 Planificación

El proyecto deberá cumplir cómo fecha de entrega máxima el 10 de julio de 2024, cumpliendo así con su meta principal.

REGISTRO DE RIESGOS

Título del Proyecto: Integración de datos institucionales heterogéneos en plataforma abierta y distribuida

Fecha: 7 junio 2024

CAMBIOS

Cambios de la versión 2.0

ID	Nombre	Responsable	Probabilidad	Impacto				Impacto
				Presup.	Planific.	Alcance	Calidad	
1								
2	Dificultades en la Integración Tecnológica	Desarrollado senior	Media	Medio	Alto	Medio	Medio	0,28
3	Resistencia del Personal a Nuevas Tecnologías	Jefe de proyecto y adminsitrador de sistemas	Media	Bajo	Medio	Medio	Medio	0,15
4	Falta de Acceso a Fuentes de Datos	Analista de datos	Media	Bajo	Alto	Alto	Alto	0,28
5	Complejidad en la Implementación de visualización de datos	Administrador de Sistemas y Analista de Datos	Media	Bajo	Medio	Bajo	Alto	0,28
6	Cambios Significativos en los Requisitos	Jefe de Proyecto	Baja	Medio	Medio	Alto	Medio	0,17
7	Desviación en Plazos de Entrega	desarrolladores	Media	Alto	Alto	Bajo	Bajo	0,28
8	Rotura del servidor	Adminsitrador de sistemas	Media	Alto	Alto	Alto	Alto	0,28
9	Rotura de sensores		Baja	Alto	Alto	Bajo	Bajo	0,17
10	Cambios funcionales en actualizaciones de herramienta usadas de terceros(Kafka,Nifi,InfluxDb)	desarrolladores	Baja	Bajo	Alto	Bajo	Medio	0,17
11	Perdida de datos almancenados	Adminsitrador de sistemas	Baja	Bajo	Bajo	Alto	Alto	0,17
12	Espacio insuficiente en el servidor	Adminsitrador de sistemas	Media	Medio	Medio	Alto	Alto	0,28
13	Problemas con la integración de APIs externas	desarrolladores	Baja	Bajo	Alto	Alto	Alto	0,17
15	Problemas con la escalabilidad de la plataforma	Adminsitrador de sistemas	Baja	Alto	Bajo	Medio	Alto	0,17
16	Problemas con la calidad de los datos integrados	Analista de datos	Media	Bajo	Bajo	Alto	Alto	0,28
17	Interrupciones debido a mantenimiento no planificado	desarrolladores	Media	Muy Bajo	Alto	Medio	Bajo	0,28
18	Retrasos en la obtención de permisos de acceso a los servicios que se consuman.	desarrolladores	Alta	Muy Bajo	Alto	Bajo	Alto	0,39

Categorización de los riesgos

Técnico:

Dificultades en la Integración Tecnológica

Complejidad en la Implementación de visualización de datos.

Falta de Acceso a Fuentes de Datos.

Rotura del servidor

Problemas con la integración de APIs externas

Problemas con la escalabilidad de la plataforma

Problemas con la calidad de los datos integrados

Interrupciones debido a mantenimiento no planificado

Rotura de sensores

Cambios funcionales en actualizaciones de herramienta usadas de terceros

Pérdida de datos almacenados

Espacio insuficiente en el servidor

Organizacional:

Desviación en Plazos de Entrega

Retrasos en la obtención de permisos de acceso

Resistencia del Personal a Nuevas Tecnologías

Retrasos en la obtención de permisos de acceso

Gestión del Proyecto:

Cambios Significativos en los Requisitos

PRESUPUESTO INICIAL. PARTIDAS DESGLOSADAS

A continuación, se muestra cada partida de coste, con las tareas que las componen, las cantidades de horas de trabajo, el precio por hora y la suma de estos subtotales por jerarquía,

I 1	*I 2*	*I 3*	*I4*	*Descripción*	*Cant*	*Unidades*	*Precio*	*Subtotal*
01				**Fase de inicio de proyecto**				
	001			Elaboración de acta de constitución del proyecto				
		01		jefe de proyecto	8	horas	60,00 €	480,00 €
	002			Revisión y aprobación de acta de constitución del proyecto (AC) con los stakeholders				
		01		jefe de proyecto	4	horas	60,00 €	240,00 €
	003			Reunión de inicio de proyecto				
		01		jefe de proyecto	4	horas	60,00 €	240,00 €
		02		Desarrollador senior	4	horas	37,50 €	150,00 €
		03		Desarrollador Junior Web	4	horas	26,25 €	105,00 €
		04		Analista de datos	4	horas	48,75 €	195,00 €
		05		Administrador de sistemas	4	horas	26,25 €	105,00 €

Tabla 15. Partida de costes - Fase de inicio de Proyecto

I 1	*I 2*	*I 3*	*I4*	*Descripción*	*Cant*	*Unidades*	*Precio*	*Subtotal*
02				**Planificación de proyecto**				

	001			Definición detallada de objetivos específicos				
		01		Desarrollador senior	8	horas	37,50 €	300,00 €
	002			Definición detallada del alcance				
		01		Desarrollador Junior Web	8	horas	26,25 €	210,00 €
	003			Captura de requisitos con el cliente				
		01		jefe de proyecto	16	horas	60,00 €	960,00 €
	004			Definir requisitos funcionales y no funcionales				
		01		Desarrollador senior	8	horas	37,50 €	300,00 €
	005			Elaboración de cronograma de hitos principales				
		01		Desarrollador senior	8	horas	37,50 €	300,00 €
	006			Estimación de presupuesto				
				Analista de datos	8	horas	48,75 €	390,00 €
	007			Reunión con el equipo y los interesados para ver los avances 1				
		01		jefe de proyecto	8	horas	60,00 €	480,00 €
		02		Desarrollador senior	8	horas	37,50 €	300,00 €
		03		Analista de datos	8	horas	48,75 €	390,00 €

		04		Administrador de sistemas	8	horas	26,25 €	210,00 €
	008			Identificación de riesgos y estrategias de mitigación				
		01		*jefe de proyecto*	*8*	horas	60,00 €	480€

Tabla 16. Partida de costes - Planificación de proyecto

I 1	***I 2***	***I 3***	***I4***	***Descripción***	***Cant***	***Unidades***	***Precio***	**subtotal**
03				**Desarrollo de arquitectura**				
	001			Definición de arquitecturas de información				
		0001		Realización de un análisis detallado de los requisitos de datos provenientes de fuentes como wifi, calefacción, etc.				
			01	Analista de datos	24	horas	48,75 €	1.170 €
			02	Administrador de sistemas	24	horas	26,25 €	630 €
		0002		Estudio de las variables a recibir de las diferentes fuentes de datos				
			01	Desarrollador Junior Web	8	horas	26,25 €	200 €
		0003		Identificación de entidades y relaciones claves específicas para cada fuente de datos				
			01	Desarrollador Junior Web	16		26,25 €	420 €
		0004		Elaboración de diagrama de flujo de datos para visualizar los procesos involucrados				
			01	Analista de datos	8	horas	48,75 €	390 €
		0005		Realizar documentación de infraestructura a utilizar				

			01	jefe de proyecto	8	horas	60,00 €	480 €
			02	Desarrollador Junior Web	8	horas	26,25 €	210 €
			03	Administrador de sistemas	8	horas	26,25 €	210 €
		0006		Estudio de cómo usar la infraestructura a utilizar				
			01	Desarrollador Junior Web	16	horas	26,25 €	420 €
	002			Estudios de las tecnologías y Plataformas				
		0001		Estudio y familiarización con la herramienta Apache NiFi y otras				
			01	Desarrollador Junior Web	16	horas	26,25 €	420 €
			02	Analista de datos	16	horas	48,75 €	780 €
			03	Administrador de sistemas	16	horas	26,25 €	420 €
		0002		Estudio y familiarización con la plataforma smartUa				
			01	Desarrollador Junior Web	16	horas	26,25 €	420 €
			02	Analista de datos	16	horas	48,75 €	780 €
			03	Administrador de sistemas	16	horas	26,25 €	420 €
		0003		Estudio de conexión con los sensores de información y la descarga de colas de mensajes MQTT				
			01	Desarrollador Junior Web	16	horas	26,25 €	420 €
		0004		conexión con los sensores de información y la descarga de mensajes				
			01	Desarrollador Junior Web	40	horas	26,25 €	1.050 €
		0005		Estudio y manejo de la OpenApi de la plataforma SmartUA				
			001	Desarrollador Junior Web	8	horas	26,25 €	210 €
		0006		Estudio de la pila ELK				

			001	Desarrollador Junior Web	8	horas	26,25 €	210 €
		0007		Estudio de la herramienta Kafka				
			001	Desarrollador Junior Web	8	horas	26,25 €	210 €
		0008		Estudio de manejo y comandos en ubuntu para desplegar tecnologías de la implementación en docker compose				
			001	Desarrollador Junior Web	8	horas	26,25 €	210 €
		00009		Pruebas de conexiones a los sensores				
			001	Desarrollador Junior Web	8	horas	26,25 €	210 €

Tabla 17. Partida de costes - Desarrollo de arquitectura

I 1	***I 2***	***I 3***	***I4***	***Descripción***	***Cant***	***Unidades***	***Precio***	***Subtotal***
04				**Diseño de la integración**				
	001			Determinar las variables que son objetos de estudio				
		01		Desarrollador senior	8	horas	37,50 €	300,00 €
	002			Creación de diagramas de arquitectura de alto nivel				
		01		Analista de datos	16	horas	48,75 €	780,00 €
	003			Desarrollo de modelo de datos detallados para cada fuentes de información				
		01		Desarrollador senior	8	horas	37,50 €	300,00 €
	004			Diseño de dashborad y gráficas para el análisis y la visualización				
		01		Desarrollador Junior Web	16	horas	26,25 €	420,00 €
	005			Identificación de patrones de diseño de la arquitectura				
		01		jefe de proyecto	16	horas	60,00 €	960,00 €

	006			Documentación de la infraestructura tecnológica				
		01		Desarrollador Junior Web	24	horas	26,25 €	630,00 €

Tabla 18. Partida de costes - Diseño de integración

I 1	*I 2*	*I 3*	*I4*	*Descripción*	*Cant*	*Unidades*	*Precio*	*Subtotal*
05				**Implantación de la plataforma**				
	001			**Configuración de la infraestructura**				
		0001		Adquisición e instalación de hardware necesario para servidores				
			01	Administrador de sistemas	40	horas	26,25 €	1.050 €
		0002		Configuración de la infraestructura				
			01	Administrador de sistemas	16	horas	26,25 €	420 €
		0003		Instalación y despliegue de herramienta				
			01	Administrador de sistemas	16	horas	26,25 €	420 €
	002			Conexiones de sensores				
		0001		Prueba de conexión con sensores y fuentes de información				
			01	Desarrollador Junior Web	16	horas	26,25 €	420 €
		0002		Prueba de obtención de los datos con herramientas como Psotman y Mosquitto.				
			01	Desarrollador Junior Web	16	horas	26,25 €	420 €
		0003		Observación detallada de la estructura de información				
			01	Desarrollador Junior Web	8	horas	26,25 €	210 €
	003			Ingesta de datos				
		0001		Realizar conexión con cada una de las fuentes de datos.				

			01	Desarrollador Junior Web	8	horas	26,25 €	210 €
		0002		Procesamiento de datos				
			01	Desarrollador Junior Web	8	horas	26,25 €	210 €
		0003		Re-estructuración de los datos				
				Desarrollador senior	8	horas	37,50 €	300 €
		0004		Enriquecimiento de datos de entrada				
			01	Desarrollador Junior Web	8	horas	26,25 €	210 €
		0005		Agregación de metadatos.				
			01	Desarrollador Junior Web	8	horas	26,25 €	210 €
	004			Análisis y visualización de datos				
		0001		Enviar datos la pila ELK				
			01	Desarrollador senior	16	horas	37,50 €	600 €
		0002		Crear índice de datos en Elastick Search				
			01	Desarrollador Junior Web	8	horas	26,25 €	210 €
		0003		Implementación de consultas DSL sobre datos				
			01	Desarrollador Junior Web	8	horas	26,25 €	210 €
		0004		Creación de dashborad con los datos que se quieres visualizar				
			01	Desarrollador senior	40	horas	37,50 €	1.500 €
		0005		Implementación de visualizaciones con Vega en Kibana				
			001	Desarrollador Junior Web	32		26,25 €	

Tabla 19. Partida de costes - Implantación de la plataforma

I 1	***I 2***	***I 3***	***I4***	***Descripción***	***Cant***	***Unidades***	***Precio***	*Subtotal*	***I 1***
06				**Pruebas y ajustes**					

	001			Ejecución de pruebas unitarias.					
		01		Desarrollador Junior Web	40	horas	26,25 €		1.050,00 €
	002			Identificación y corrección de análisis en el código					
		01		Desarrollador Junior Web	8	horas	26,25 €		210,00 €
	003			Validación de funcionalidad y cumplimiento de requisitos					
		01		Desarrollador senior	8	horas	37,50 €		300,00 €
	004			Integración de datos desde diversas fuentes en el entorno de prueba					
		01		Desarrollador senior	40	horas	37,50 €		1.500,00 €
	005			Ejecución de pruebas para evaluar la coherencia de la información					
		01		Analista de datos	16	horas	48,75 €		780,00 €

Tabla 20. Partida de costes - Prueba y ajustes

I 1	***I 2***	***I 3***	***I4***	***Descripción***	***Cant***	***Unidades***	***Precio***	***Subtotal***
07				**Revisión y cierre del proyecto**				

	001			Revisión de la planificación realizada				
		01		jefe de proyecto	8	horas	60,00 €	480,00 €
	002			Monitoreo y evaluación de la implementación operativa				
		01		Administrador de sistemas	40	horas	26,25 €	1.050,00 €
	003			Reunión de revisión y aprobación por partes interesadas				
		01		jefe de proyecto	8	horas	60,00 €	480,00 €
	004			Documentación final y entrega al cliente				
		01		jefe de proyecto	24	horas	60,00 €	1.440,00 €
	005			Evaluación de satisfacción de los stakeholders				
		01		jefe de proyecto	8	horas	60,00 €	480,00 €

Tabla 21. Partida de costes - Revisión y cierre de proyecto

CIERRE DEL PROYECTO

Planificación final

A continuación se muestra la planificación final del proyecto, resultante del seguimiento y actualización de la planificación inicial por bloque para una visión detallada de cada etapa.

EDT	Nombre de tarea	Comienzo	Fin	% completado
1	**Fase de inicio de proyecto**	**lun 23/10/23**	**vie 27/10/23**	**100%**
1.1	Elaboración de acta de constitución del pr	lun 23/10/23	lun 23/10/23	100%
1.2	Entrega de acta de constitución de proyec	mié 25/10/23	mié 25/10/23	100%
1.3	Revisión y aprobación de acta de constitu	jue 26/10/23	jue 26/10/23	100%
1.4	Reunión de inicio de proyecto	vie 27/10/23	vie 27/10/23	100%

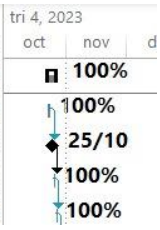


Ilustración 50. Planificación final. Inicio de proyecto

EDT	Nombre de tarea	Comienzo	Fin	% completado
2	**Planificación de proyecto**	**vie 27/10/23**	**mié 08/11/23**	**100%**
2.1	Definición detallada de objetivos específic	vie 27/10/23	lun 30/10/23	100%
2.2	Definición alcance	lun 30/10/23	lun 30/10/23	100%
2.3	Captura de requisitos iniciales	mar 31/10/23	mar 31/10/23	100%
2.4	Definir requisitos funcionales y no funcion	jue 02/11/23	jue 02/11/23	100%
2.5	Elaboración de cronograma de hitos princ	vie 03/11/23	vie 03/11/23	100%
2.6	Estimación de presupuesto	lun 06/11/23	lun 06/11/23	100%
2.7	Entrega de planificación preliminar	lun 06/11/23	lun 06/11/23	100%
2.8	**Reunión con el equipo y los interesados**	**mar 07/11/23**	**mar 07/11/23**	**100%**
2.8.1	Reunión con el equipo y los interesados	mar 07/11/23	mar 07/11/23	100%
2.9	Identificación de riesgos y estrategias de r	mié 08/11/23	mié 08/11/23	100%
2.10	Entrega de plan de riesgos	mié 08/11/23	mié 08/11/23	100%

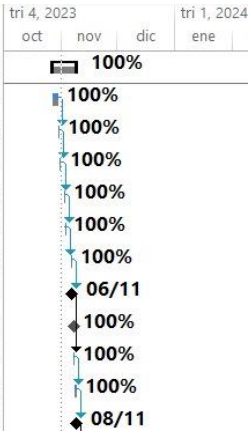


Ilustración 51. Planificación final de Planificación

EDT	Nombre de tarea	Comienzo	Fin	Nombres de los recursos
3	**Desarrollo de arquitectura**	**jue 09/11/23**	**vie 22/03/24**	
3.1	**Arquitectura de proyecto**	**jue 09/11/23**	**mar 19/12/23**	
3.1.1	Realización de un análisis detallado de l	jue 09/11/23	mié 22/11/23	Administrador de sistem
3.1.2	Identificación de la necesida de infraest	jue 23/11/23	vie 24/11/23	Desarrollador Junior We
3.1.3	Identificación de entidades y relaciones	vie 24/11/23	vie 24/11/23	Desarrollador Junior We
3.1.4	Estudio de alternativas de tecnologías	mar 28/11/23	jue 30/11/23	Analista de datos
3.1.5	Definir arquitectura o marco de integra	mié 29/11/23	jue 14/12/23	Administrador de sistem
3.1.6	Selección de las tecnologías	vie 15/12/23	mar 19/12/23	Desarrollador Junior We
3.1.7	Entrega de la infraestuctura a utilizar	lun 18/12/23	lun 18/12/23	
3.2	**Estudios de las tecnologías y Plataforma**	**mar 19/12/23**	**vie 22/03/24**	
3.2.1	Estudio y familiarización con la herrami	mar 19/12/23	jue 21/12/23	Administrador de sistem
3.2.2	Estudio y familiarización con la platafor	mar 12/03/24	mar 12/03/24	Administrador de sistem
3.2.3	Estudio de conexión con los sensores de	mié 13/03/24	jue 14/03/24	Desarrollador Junior We
3.2.4	Estudio y manejo de la openApi de smar	vie 15/03/24	vie 15/03/24	Desarrollador Junior We
3.2.5	Estudio y familiarización te tecnologías	vie 15/03/24	lun 18/03/24	Desarrollador Junior We
3.2.6	Documentar tecnologías	lun 18/03/24	mar 19/03/24	Desarrollador Junior We
3.2.7	Estudio de manejo y comandos en ubun	mar 19/03/24	mié 20/03/24	Desarrollador Junior We
3.2.8	Pruebas de tecnologías	mié 20/03/24	jue 21/03/24	Desarrollador Junior We
3.2.9	Diagrama de infraestructura tecnológica	jue 21/03/24	vie 22/03/24	Desarrollador Junior We

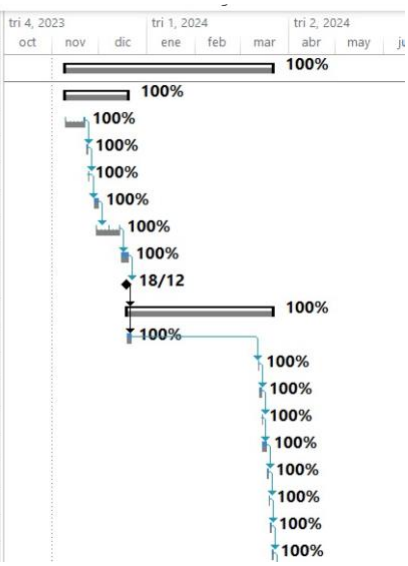


Ilustración 52. Planificación final Desarrollo de la Arquitectura

EDT	Nombre de tarea	Comienzo	Fin	Nombres de los recursos
4	**Analisis y Diseño de la integración**	**vie 22/03/24**	**lun 01/04/24**	
4.1	Determinar las variables que son objetos	vie 22/03/24	lun 25/03/24	Desarrollador senior
4.2	Identifcar actores	lun 25/03/24	lun 25/03/24	Analista de datos
4.3	Definir procesamiento de dato para cada	mar 26/03/24	mar 26/03/24	Analista de datos
4.4	Establecer requisitos	mié 27/03/24	mié 27/03/24	Desarrollador senior
4.5	Plan de pruebas	jue 28/03/24	vie 29/03/24	Desarrollador Junior We
4.6	Diagramas de casos de uso y espacificacic	vie 29/03/24	lun 01/04/24	Desarrollador Junior We
4.7	Agregar de la documentación de diseño a	lun 01/04/24	lun 01/04/24	

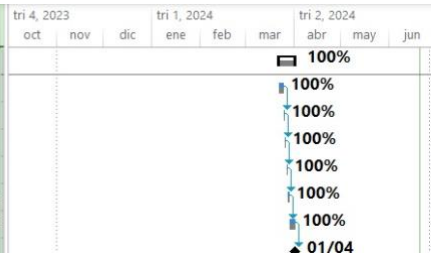


Ilustración 53. Planificación final Análisis y Diseño de la integración

EDT	Nombre de tarea	Comienzo	Fin	Nombres de los recursos
5	**Implantación de la plataforma**	**mar 02/04/24**	**vie 03/05/24**	
5.1	**Configuración de infraestrucuta**	**mar 02/04/24**	**lun 08/04/24**	
5.1.1	Creación de diagramas de despliegue	mar 02/04/24	mar 02/04/24	Administrador de sistem
5.1.2	Configuración de la infraestructura	mié 03/04/24	jue 04/04/24	Administrador de sistem
5.1.3	Instalación y despliegue de herramienta	vie 05/04/24	lun 08/04/24	Administrador de sistem
5.2	**Conexión de sensores**	**lun 08/04/24**	**mar 09/04/24**	
5.2.1	Prueba de conexión con sensores y fuer	lun 08/04/24	lun 08/04/24	Desarrollador Junior We
5.2.2	Prueba de obtención de los datos con h	mar 09/04/24	mar 09/04/24	Desarrollador Junior We
5.2.3	Observación detallada de la estructura	mar 09/04/24	mar 09/04/24	Desarrollador Junior We
5.3	**Ingesta de datos de las diversas fuentes**	**mié 10/04/24**	**mar 23/04/24**	
5.3.1	Realizar conexión con cada una de las f	mié 10/04/24	mié 10/04/24	Desarrollador Junior We
5.3.2	Procesamiento de datos	jue 11/04/24	lun 15/04/24	Desarrollador Junior We
5.3.3	Re-estructuración de la infromación de	mar 16/04/24	mié 17/04/24	Desarrollador Junior We
5.3.4	Enriquecimiento de datos de entrada.	jue 18/04/24	vie 19/04/24	Desarrollador Junior We
5.3.5	Agregación de meta-datos.	vie 19/04/24	mar 23/04/24	Desarrollador Junior We
5.4	**Almacenamiento de datos**	**lun 22/04/24**	**vie 03/05/24**	
5.4.1	Envío de datos a la plataforma kunna	mar 23/04/24	mié 24/04/24	Desarrollador senior
5.4.2	**Análisis y visualización de datos**	**jue 25/04/24**	**vie 03/05/24**	
5.4.2.1	Diseñar visualización	jue 25/04/24	jue 25/04/24	Desarrollador senior
5.4.2.2	Implementar visualización	jue 25/04/24	mar 30/04/24	Desarrollador Junior We
5.4.2.3	Monitorear carga de datos	mar 30/04/24	mié 01/05/24	Analista de datos
5.4.2.4	Creación de dashboard con los datos	mié 01/05/24	vie 03/05/24	Desarrollador senior
5.4.3	Integración efectiva de los datos	lun 22/04/24	lun 22/04/24	
6	**Pruebas y ajustes**	**mar 23/04/24**	**mar 18/06/24**	
7	**Revisión y cierre del proyecto**	**mar 18/06/24**	**mié 26/06/24**	

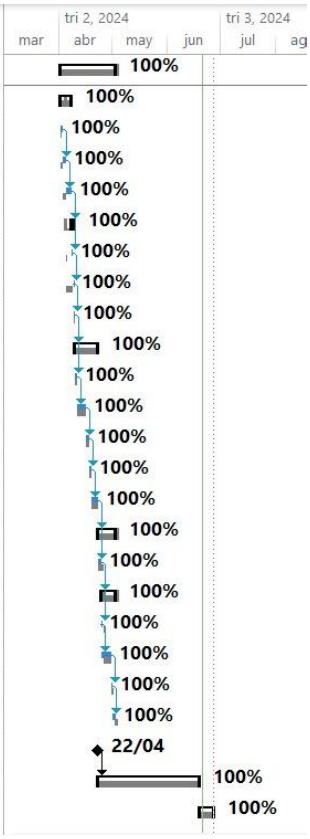


Ilustración 54. Planificación final Implantación de la plataforma

EDT	Nombre de tarea	Comienzo	Fin	Nombres de los recursos
6	**Pruebas y ajustes**	**mar 23/04/24**	**mar 18/06/24**	
6.1	Ejecución de pruebas de rendimiento.	mar 23/04/24	mar 23/04/24	Desarrollador Junior We
6.2	Identificación y corrección de errores	mar 11/06/24	mar 11/06/24	Desarrollador Junior We
6.3	Validación de funcionalidad y cumplimient	mié 12/06/24	mié 12/06/24	Desarrollador senior
6.4	Ejecución de pruebas de extremo a extren	mar 18/06/24	mar 18/06/24	Analista de datos
7	**Revisión y cierre del proyecto**	**mar 18/06/24**	**mié 26/06/24**	

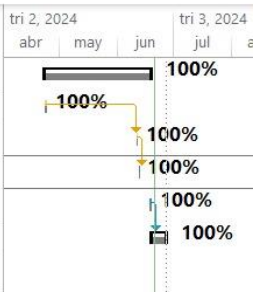


Ilustración 55. Planificación final Pruebas y ajustes

EDT	Nombre de tarea	Comienzo	Fin	Nombres de los recursos
7	**Revisión y cierre del proyecto**	**mar 18/06/24**	**mié 26/06/24**	
7.1	Entrega de presupuesto final del proyecto	mar 18/06/24	mar 18/06/24	
7.2	**Revisión de la planificación realizada**	**jue 20/06/24**	**lun 24/06/24**	
7.2.1	Monitoreo y evaluación de la implemen	jue 20/06/24	lun 24/06/24	Administrador de sistem
7.3	Reunión de revisión	mié 26/06/24	mié 26/06/24	Desarrollador Junior We
7.4	Documentación final y entrega	mié 26/06/24	mié 26/06/24	Administrador de sistem
7.5	Presentación del proyecto	mié 26/06/24	mié 26/06/24	

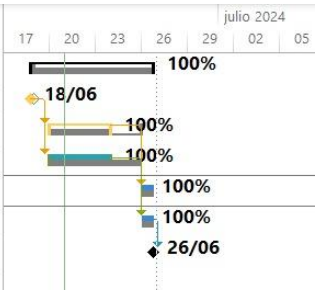


Ilustración 56. Planificación final Revisión y cierre de proyecto

Presupuesto final. Partidas desglosadas

A continuación, se muestra cada partida de coste, con las tareas que las componen, las cantidades de horas de trabajo, el precio por hora y la suma de estos subtotales por jerarquía.

Fase de inicio de proyecto

I 1	*I 2*	*I 3*	*I 4*	*Descripción*	*Cantidad*	*Unidades*	*Precio*	*Subtotal (4)*	*Subtotal (3)*	*Subtotal (2)*
01				**Fase de inicio de proyecto**						
	001			Elaboración de acta de constitución del proyecto						**480,00 €**
		01		jefe de proyecto	8	horas	60,00 €		480,00 €	
	002			Revisión y aprobación de acta de constitución del proyecto (AC) con los stakeholders						**240,00 €**
		01		jefe de proyecto	4	horas	60,00 €		240,00 €	
	003			Reunión de inicio de proyecto						**795,00 €**
		01		jefe de proyecto	4	horas	60,00 €		240,00 €	
		02		Desarrollador senior	4	horas	37,50 €		150,00 €	
		03		Desarrollador Junior Web	4	horas	26,25 €		105,00 €	
		04		Analista de datos	4	horas	48,75 €		195,00 €	
		05		Administrador de sistemas	4	horas	26,25 €		105,00 €	

Tabla 22. Presupuesto final Fase de inicio de proyecto

Planificación de proyecto

I 1	*I 2*	*I 3*	*I 4*	*Descripción*	*Cantidad*	*Unidades*	*Precio*	*Subtotal (4)*	*Subtotal (3)*
'02				**Planificación de proyecto**					
	001			Definición detallada de objetivos específicos					
		01		Desarrollador senior	8	horas	37,50 €		300,00 €
	002			Definición detallada del alcance					
		01		Desarrollador Junior Web	8	horas	26,25 €		210,00 €
	003			Captura de requisitos iniciales					
		01		jefe de proyecto	16	horas	60,00 €		960,00 €
	004			Definir requisitos funcionales y no funcionales					
		01		Desarrollador senior	8	horas	37,50 €		300,00 €
	005			Elaboración de cronograma de hitos principales					
		01		Desarrollador senior	4	horas	37,50 €		150,00 €
	006			Estimación de presupuesto					
				Analista de datos	8	horas	48,75 €		390,00 €

	007			Reunión con el equipo y los interesados para ver los avances 1					
		01		jefe de proyecto	8	horas	60,00 €		480,00 €
		02		Desarrollador senior	8	horas	37,50 €		300,00 €
		03		Analista de datos	8	horas	48,75 €		390,00 €
		04		Administrador de sistemas	8	horas	26,25 €		210,00 €
	008			Identificación de riesgos y estrategias de mitigación					
		01		*jefe de proyecto*	*8*	horas	60,00 €		480,00 €

Tabla 23. Presupuesto final Planificación de proyecto

Desarrollo de arquitectura

I 1	***I 2***	***I 3***	***I4***	***Descripción***	***Cantidad***	***Unidades***	***Precio***	***Subtotal (4)***	***Subtotal (3)***
03				**Desarrollo de arquitectura**					
	001			Arquitectura de proyecto					
		0001		Realización de un análisis detallado de los requisitos de datos provenientes de fuentes como wifi, calefacción, etc.					1.500,00 €
			01	Analista de datos	20	horas	48,75 €	975 €	
			02	Administrador de sistemas	20	horas	26,25 €	525 €	

		0002		Identificación de la necesidad de infraestructura tecnológica					315,00 €
			01	Desarrollador Junior Web	12	horas	26,25 €	315 €	
		0003		Identificación de entidades y relaciones claves específicas para cada fuente de datos					105,00 €
			01	Desarrollador Junior Web	4		26,25 €	105 €	
		0004		Estudio de alternativas de tecnologías					1.170,00 €
			01	Analista de datos	24	horas	48,75 €	1.170 €	
		0005		Definir arquitectura o marco de integración					900,00 €
			01	jefe de proyecto	8	horas	60,00 €	480 €	
			02	Desarrollador Junior Web	8	horas	26,25 €	210 €	
			03	Administrador de sistemas	8	horas	26,25 €	210 €	
		0006		Selección de las tecnologías					630,00 €
			01	Desarrollador Junior Web	24	horas	26,25 €	630 €	
	002			Estudios de las tecnologías y Plataformas					
		0001		Estudio y familiarización con la herramienta de captura y transformación de los datos					2.430,00 €
			01	Desarrollador Junior Web	24	horas	26,25 €	630 €	

			02	Analista de datos	24	horas	48,75 €	1.170 €	
			03	Administrador de sistemas	24	horas	26,25 €	630 €	
		0002		Estudio y familiarización con la plataforma kunna					101,25 €
			01	Desarrollador Junior Web	4	horas	26,25 €	105 €	
			02	Analista de datos	4	horas	48,75 €	195 €	
			03	Administrador de sistemas	4	horas	26,25 €	105 €	
		0003		Estudio de conexión con los sensores de información y la descarga de colas de mensajes MQTT					420,00 €
			01	Desarrollador Junior Web	16	horas	26,25 €	420 €	
		0004		Estudio y manejo de la OpenApi de la plataforma SmartUA					105,00 €
			001	Desarrollador Junior Web	4	horas	26,25 €	105 €	
		0005		Estudio y familiarización con las herramientas de visualización					210,00 €
			001	Desarrollador Junior Web	8	horas	26,25 €	210 €	
		0006		Documentar tecnologías					210,00 €
			001	Desarrollador Junior Web	8	horas	26,25 €	210 €	
		0007		Estudio de manejo y comandos en ubuntu para desplegar tecnologías					210,00 €

				de la implementación en docker compose					
			001	Desarrollador Junior Web	8	horas	26,25 €	210 €	
		0008		Pruebas de tecnologías					210,00 €
			001	Desarrollador Junior Web	8	horas	26,25 €	210 €	0,00 €
		0009							0,00 €
			001	Diagrama de infraestructura tecnológica					210,00 €
				Desarrollador Junior Web	8	horas	26,25 €	210 €	0,00 €

Tabla 24. Presupuesto final Desarrollo de arquitectura

Análisis y Diseño de la integración

I 1	***I 2***	***I 3***	***I4***	***Descripción***	***Cantidad***	***Unidades***	***Precio***
04				**Análisis y Diseño de la integración**			
	001			Determinar las variables que son objetos de estudio			
		01		Desarrollador senior	8	horas	37,50 €
	002			Identificar actores			
		01		Analista de datos	4	horas	48,75 €
	003			Definir procesamiento para cada fuente de datos			
		01		Desarrollador senior	8	horas	37,50 €
	004			Establecer requisitos			
		01		Desarrollador Junior Web	8	horas	26,25 €
	005			Plan de pruebas			

		01		Desarrollador Junior Web	12	horas	26,25 €
	006			Diagramas de casos de uso y especificaciones			
		01		Desarrollador Junior Web	12	horas	26,25 €

Tabla 25. Presupuesto final Análisis y Diseño de la integración

Implantación de la plataforma

I 1	***I 2***	***I 3***	***I 4***	***Descripción***	***Cantidad***	***Unidades***	***Precio***	***Subtotal (4)***	***Subtotal (3)***
05				**Implantación de la plataforma**					
	001			**Configuración de la infraestructura**					
		0001		Creación de diagrama de despliegue					210,00 €
			01	Administrador de sistemas	8	horas	26,25 €	210 €	
		0002		Configuración de la infraestructura					420,00 €
			01	Administrador de sistemas	16	horas	26,25 €	420 €	
		0003		Instalación y despliegue de herramienta					315,00 €
			01	Administrador de sistemas	12	horas	26,25 €	315 €	
	002			Conexiones de sensores					
		0001		Prueba de conexión con sensores y fuentes de información					105,00 €
			01	Desarrollador Junior Web	4	horas	26,25 €	105 €	

		0002		Prueba de obtención de los datos con herramientas como Psotman y Mosquitto.					105,00 €
			01	Desarrollador Junior Web	4	horas	26,25 €	105 €	
		0003		Observación detallada de la estructura de información					105,00 €
			01	Desarrollador Junior Web	4	horas	26,25 €	105 €	
	003			Ingesta de datos					
		0001		Realizar conexión con cada una de las fuentes de datos.					210,00 €
			01	Desarrollador Junior Web	8	horas	26,25 €	210 €	
		0002		Procesamiento de datos					630,00 €
			01	Desarrollador Junior Web	24	horas	26,25 €	630 €	
		0003		Re-estructuración de los datos					600,00 €
				Desarrollador senior	16	horas	37,50 €	600 €	
		0004		Enriquecimiento de datos de entrada					315,00 €
			01	Desarrollador Junior Web	12	horas	26,25 €	315 €	
		0005		Agregación de metadatos.					420,00 €
			01	Desarrollador Junior Web	16	horas	26,25 €	420 €	
	004			Análisis y visualización de datos					

		0001		Diseñar visualización					150,00 €
			01	Desarrollador senior	4	horas	37,50 €	150 €	
		0002		Implementar visualización					630,00 €
			01	Desarrollador Junior Web	24	horas	26,25 €	630 €	
		0003		Monitorear carga de datos					315,00 €
			01	Desarrollador Junior Web	12	horas	26,25 €	315 €	
		0004		Creación de dashboard					600,00 €
			01	Desarrollador senior	16	horas	37,50 €	600 €	

Tabla 26. Presupuesto final Implantación de la plataforma

Pruebas y ajustes

I 1	***I 2***	***I 3***	***I 4***	***Descripción***	***Cantidad***	***Unidades***	***Precio***	***Subtotal (4)***	***Subtotal (3)***	***Subtotal (2)***
06				**Pruebas y ajustes**						
	001			ejecución de pruebas de rendimiento						**1.050,00 €**
		01		Desarrollador Junior Web	8	horas	26,25 €		210,00 €	
	002			Identificación y corrección de errores						**210,00 €**
		01		Desarrollador Junior Web	4	horas	26,25 €		105,00 €	
	003			Validación de funcionalidad y						**300,00 €**

				cumplimiento de requisitos						
		01		Desarrollador senior	4	horas	37,50 €		150,00 €	

Tabla 27. Presupuesto final Pruebas y ajustes

Revisión y cierre del proyecto

I 1	***I 2***	***I 3***	***I 4***	***Descripción***	***Cantidad***	***Unidades***	***Precio***	***Subtotal (4)***	***Subtotal (3)***	***Subtotal (2)***
07				**Revisión y cierre del proyecto**						
	001			Revisión de la planificación realizada						**480,00 €**
		01		jefe de proyecto	8	horas	60,00 €		480,00 €	
	002			Monitoreo y evaluación de la implementación operativa						**485,63 €**
		01		Administrador de sistemas	18,5	horas	26,25 €		485,63 €	
	003			Reunión de revisión y aprobación						**480,00 €**
		01		jefe de proyecto	8	horas	60,00 €		480,00 €	
	004			Documentación final y entrega						**1.440,00 €**
		01		jefe de proyecto	8	horas	60,00 €		480,00 €	

Tabla 28. Presupuesto final Revisión y cierre del proyecto

DESPLIEGUE DE APACHE NIFI

```
version: "3"
services:
apache-nifi:
image: apache/nifi:latest
container_name: apache-nifi
hostname: apache-nifi
#shm_size: 2G
security_opt:
- seccomp:unconfined
environment:
- TZ=Europe/Madrid
#- NIFI_WEB_HTTP_PORT=8080
#- NIFI_WEB_HTTPS_PORT=8443
- NIFI_WEB_PROXY_HOST=156.35.98.30:8443
- SINGLE_USER_CREDENTIALS_USERNAME=admin
- SINGLE_USER_CREDENTIALS_PASSWORD=*********
- NIFI_JVM_HEAP_INIT=512m
- NIFI_JVM_HEAP_MAX=1g

- AIRWAVE_API_USERNAME=aruba_API
- AIRWAVE_API_PASSWORD=4+u+E;M2z.
- AIRWAVE_API_BASE_URL=https://airwave.uniovi.es:443
volumes:
- "./volumes/opt/nifi/nifi-current/conf:/opt/nifi/nifi-current/conf"
- "./volumes/opt/nifi/nifi-current/database_repository:/opt/nifi/nifi-
current/database_repository"
- "./volumes/opt/nifi/nifi-current/flowfile_repository:/opt/nifi/nifi-
current/flowfile_repository"
- "./volumes/opt/nifi/nifi-current/content_repository:/opt/nifi/nifi-
current/content_repository"
- "./volumes/opt/nifi/nifi-current/provenance_repository:/opt/nifi/nifi-
current/provenance_repository"
- "./volumes/opt/nifi/nifi-current/state:/opt/nifi/nifi-current/state"
- "./volumes/opt/nifi/nifi-current/logs:/opt/nifi/nifi-current/logs"
- "./volumes/opt/airwave:/opt/airwave"
ports:
- "8080:8080"    # nifi.web.http.port
- "8443:8443"    # nifi.web.https.port
- "10000:10000"  # nifi.remote.input.socket.port
- "8000:8000"    # java.arg.debug
```

```
version: "3.3"

#volumes:
# certs:
#   driver: local
# esdata01:
#   driver: local
# kibanadata:
#   driver: local
# metricbeatdata01:
#   driver: local
# filebeatdata01:
#   driver: local
# logstashdata01:
#   driver: local

#networks:
# default:
#   name: elastic
#   external: false


services:
setup:
image: docker.elastic.co/elasticsearch/elasticsearch:${STACK_VERSION}
volumes:
-
"./volumes/usr/share/elasticsearch/config/certs:/usr/share/elasticsearch/confi
g/certs"
user: "0"
command: >
bash -c '
if [ x${ELASTIC_PASSWORD} == x ]; then
echo "Set the ELASTIC_PASSWORD environment variable in the .env file";
exit 1;
elif [ x${KIBANA_PASSWORD} == x ]; then
```

```
echo "Set the KIBANA_PASSWORD environment variable in the .env file";
exit 1;
fi;
if [ ! -f config/certs/ca.zip ]; then
echo "Creating CA";
bin/elasticsearch-certutil ca --silent --pem -out config/certs/ca.zip;
unzip config/certs/ca.zip -d config/certs;
fi;
if [ ! -f config/certs/certs.zip ]; then
echo "Creating certs";
echo -ne \
"instances:\n"\
"  - name: es01\n"\
"    dns:\n"\
"      - es01\n"\
"      - localhost\n"\
"    ip:\n"\
"      - 127.0.0.1\n"\
"  - name: kibana\n"\
"    dns:\n"\
"      - kibana\n"\
"      - localhost\n"\
"    ip:\n"\
"      - 127.0.0.1\n"\
> config/certs/instances.yml;
bin/elasticsearch-certutil cert --silent --pem -out config/certs/certs.zip --
in config/certs/instances.yml --ca-cert config/certs/ca/ca.crt --ca-key
config/certs/ca/ca.key;
unzip config/certs/certs.zip -d config/certs;
fi;
echo "Setting file permissions"
chown -R root:root config/certs;
find . -type d -exec chmod 750 \{\} \;;
find . -type f -exec chmod 640 \{\} \;;
echo "Waiting for Elasticsearch availability";
until curl -s --cacert config/certs/ca/ca.crt https://es01:9200 | grep -q
"missing authentication credentials"; do sleep 30; done;
echo "Setting kibana_system password";
```

```
- xpack.license.self_generated.type=${LICENSE}
#mem_limit: ${ES_MEM_LIMIT}
ulimits:
memlock:
soft: -1
hard: -1
healthcheck:
test:
[
"CMD-SHELL",
"curl -s --cacert config/certs/ca/ca.crt https://localhost:9200 | grep -q 'missing
authentication credentials'",
]
interval: 10s
timeout: 10s
retries: 120
depends_on:
- setup
restart: unless-stopped

kibana:
image: docker.elastic.co/kibana/kibana:${STACK_VERSION}
labels:
co.elastic.logs/module: kibana
volumes:
- "./volumes/usr/share/elasticsearch/config/certs:/usr/share/kibana/config/certs"
- "./volumes/usr/share/kibana/data:/usr/share/kibana/data"
ports:
- ${KIBANA_PORT}:5601
environment:
- SERVERNAME=kibana
- ELASTICSEARCH_HOSTS=https://es01:9200
- ELASTICSEARCH_USERNAME=kibana_system
- ELASTICSEARCH_PASSWORD=${KIBANA_PASSWORD}
- ELASTICSEARCH_SSL_CERTIFICATEAUTHORITIES=config/certs/ca/ca.crt
- XPACK_SECURITY_ENCRYPTIONKEY=${ENCRYPTION_KEY}
- XPACK_ENCRYPTEDSAVEDOBJECTS_ENCRYPTIONKEY=${ENCRYPTION_KEY}
- XPACK_REPORTING_ENCRYPTIONKEY=${ENCRYPTION_KEY}
```

```
until curl -s -X POST --cacert config/certs/ca/ca.crt -u
"elastic:${ELASTIC_PASSWORD}" -H "Content-Type: application/json"
https://es01:9200/_security/user/kibana_system/_password -d
"{\"password\":\"${KIBANA_PASSWORD}\"}" | grep -q "^{}"; do sleep 10; done;
echo "All done!";
'
healthcheck:
test: ["CMD-SHELL", "[ -f config/certs/es01/es01.crt ]"]
interval: 1s
timeout: 5s
retries: 120

es01:
image: docker.elastic.co/elasticsearch/elasticsearch:${STACK_VERSION}
labels:
co.elastic.logs/module: elasticsearch
volumes:
-
"./volumes/usr/share/elasticsearch/config/certs:/usr/share/elasticsearch/config/ce
rts"
- "./volumes/usr/share/elasticsearch/data:/usr/share/elasticsearch/data"
ports:
- ${ES_PORT}:9200
environment:
- node.name=es01
- cluster.name=${CLUSTER_NAME}
- discovery.type=single-node
- ELASTIC_PASSWORD=${ELASTIC_PASSWORD}
- bootstrap.memory_lock=true
- xpack.security.enabled=true
- xpack.security.http.ssl.enabled=true
- xpack.security.http.ssl.key=certs/es01/es01.key
- xpack.security.http.ssl.certificate=certs/es01/es01.crt
- xpack.security.http.ssl.certificate_authorities=certs/ca/ca.crt
- xpack.security.transport.ssl.enabled=true
- xpack.security.transport.ssl.key=certs/es01/es01.key
- xpack.security.transport.ssl.certificate=certs/es01/es01.crt
- xpack.security.transport.ssl.certificate_authorities=certs/ca/ca.crt
- xpack.security.transport.ssl.verification_mode=certificate
```

```
- SERVER_PUBLICBASEURL=http://156.35.98.30:5601
#mem_limit: ${KB_MEM_LIMIT}
healthcheck:
test:
[
"CMD-SHELL",
"curl -s -I http://localhost:5601 | grep -q 'HTTP/1.1 302 Found'",
]
interval: 10s
timeout: 10s
retries: 120
depends_on:
- es01
restart: unless-stopped

metricbeat01:
image: docker.elastic.co/beats/metricbeat:${STACK_VERSION}
user: root
volumes:
- "./volumes/usr/share/elasticsearch/config/certs:/usr/share/metricbeat/certs"
- "./volumes/usr/share/metricbeat/data:/usr/share/metricbeat/data"
-
"./volumes/usr/share/metricbeat/metricbeat.yml:/usr/share/metricbeat/metricbeat
.yml:ro"
- "/var/run/docker.sock:/var/run/docker.sock:ro"
- "/sys/fs/cgroup:/hostfs/sys/fs/cgroup:ro"
- "/proc:/hostfs/proc:ro"
- "/:/hostfs:ro"
environment:
- ELASTIC_USER=elastic
- ELASTIC_PASSWORD=${ELASTIC_PASSWORD}
- ELASTIC_HOSTS=https://es01:9200
- KIBANA_HOSTS=http://kibana:5601
- LOGSTASH_HOSTS=http://logstash01:9600
depends_on:
- es01
- kibana
restart: unless-stopped
```

```
filebeat01:
       image: docker.elastic.co/beats/filebeat:${STACK_VERSION}
       user: root
       volumes:
       - "./volumes/usr/share/elasticsearch/config/certs:/usr/share/filebeat/certs"
       - "./volumes/usr/share/filebeat/data:/usr/share/filebeat/data"
       - "./volumes/usr/share/filebeat/ingest_data:/usr/share/filebeat/ingest_data"
       #     -
       "./volumes/usr/share/filebeat/filebeat.yml:/usr/share/filebeat/filebeat.yml:r
       o"
       - "/var/lib/docker/containers:/var/lib/docker/containers:ro"
       - "/var/run/docker.sock:/var/run/docker.sock:ro"
       environment:
       - ELASTIC_USER=elastic
       - ELASTIC_PASSWORD=${ELASTIC_PASSWORD}
       - ELASTIC_HOSTS=https://es01:9200
       - KIBANA_HOSTS=http://kibana:5601
       - LOGSTASH_HOSTS=http://logstash01:9600
       depends_on:
       - es01
       restart: unless-stopped

       logstash01:
       image: docker.elastic.co/logstash/logstash:${STACK_VERSION}
       labels:
       co.elastic.logs/module: logstash
       user: root
       volumes:
       - "./volumes/usr/share/elasticsearch/config/certs:/usr/share/logstash/certs"
       - "./volumes/usr/share/logstash/data:/usr/share/logstash/data"
       - "./volumes/usr/share/filebeat/ingest_data:/usr/share/logstash/ingest_data"
       - - "./volumes/usr/share/logstash/pipeline:/usr/share/logstash/pipeline:ro"
           -
       "./volumes/usr/share/logstash/pipelines.yml:/usr/share/logstash/pipelines.yml
       :ro"

       environment:
       - xpack.monitoring.enabled=false
```

```
      - ELASTIC_USER=elastic
      - ELASTIC_PASSWORD=${ELASTIC_PASSWORD}
      - ELASTIC_HOSTS=https://es01:9200
      depends_on:
      - es01
      - kibana
      restart: unless-stopped
```

LOGSTASH PIPELINES

pipeline-wifi.conf

```
input {
heartbeat { interval => 2500
add_field => { "pipeline" => "wifi" }
}

}

filter {
if [pipeline] == "wifi" {
ruby {
code => "
require 'net/http'
require 'uri'
require 'time'
require 'json'
time_end = Time.now.utc
time_start = time_end - (30 * 60) # Resta 30 minutos en segundos
#       time_start = time_end - ( * 60 * 60) # Resta 24 horas en segundos
time_start_str = time_start.strftime('%Y-%m-%dT%H:%M:%SZ')
time_end_str = time_end.strftime('%Y-%m-%dT%H:%M:%SZ')

puts 'time_start:wifi ' + time_start_str
puts 'time_end:wifi ' + time_end_str
time_end = Time.now.utc
time_start = (time_end - 1800).strftime('%Y-%m-%dT%H:%M:%SZ')

uri =
URI('https://openapi.smartua.es/data/eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpYXQiOjE2O
TkzMDg1MzN9.vWosk2gb_jdX0Zt2mpwknOxkGYQ20E5nDbUwKvKPFic')
http = Net::HTTP.new(uri.host, uri.port)
http.use_ssl = true

request = Net::HTTP::Post.new(uri.path, 'Content-Type' => 'application/json')
request.body = {
'time_start' => time_start_str,
'time_end' => time_end_str
}.to_json

response = http.request(request)
data = JSON.parse(response.body)

# Aquí procesa los datos según sea necesario

event.set('api_response', data) # Agrega la respuesta de la API a los datos del evento
```

pipeline-loRa.conf

```
input {
heartbeat { interval => 3000
add_field => { "pipeline" => "lora"
}
}
}


filter {
if [pipeline] == "lora" {
ruby {
code => "
require 'net/http'
require 'uri'
require 'time'
require 'json'
time_end = Time.now.utc
time_start = time_end - (30 * 60) # Resta 30 minutos en segundos
#       time_start = time_end - ( * 60 * 60) # Resta 24 horas en segundos
time_start_str = time_start.strftime('%Y-%m-%dT%H:%M:%SZ')
time_end_str = time_end.strftime('%Y-%m-%dT%H:%M:%SZ')

puts 'time_start:lora ' + time_start_str
puts 'time_end lora: ' + time_end_str
time_end = Time.now.utc
time_start = (time_end - 1800).strftime('%Y-%m-%dT%H:%M:%SZ')

uri =
URI('https://openapi.smartua.es/data/eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpYXQiOjE2
OTkzMDg2NTR9.15exbHhkQV0kZyFHjyJ_TmrO1lDi06VB8qr9bveyV40')
http = Net::HTTP.new(uri.host, uri.port)
http.use_ssl = true

request = Net::HTTP::Post.new(uri.path, 'Content-Type' => 'application/json')
request.body = {
'time_start' => time_start_str,
'time_end' => time_end_str
}.to_json

response = http.request(request)
data = JSON.parse(response.body)

# Aquí procesa los datos según sea necesario

event.set('api_response',data) # Agrega la respuesta de la API a los datos del evento
"
}
}
}

output {
if [pipeline] == "lora" {
elasticsearch {
index => "lora"
hosts=> "${ELASTIC_HOSTS}"
user=> "${ELASTIC_USER}"
password=> "${ELASTIC_PASSWORD}"
cacert=> "certs/ca/ca.crt"
}
}
```

Prueba de integración

En este anexo se adjuntan algunas trazas de las pruebas de integración para las dos fuentes de datos implementadas.

Puntos de Acceso de Wifi

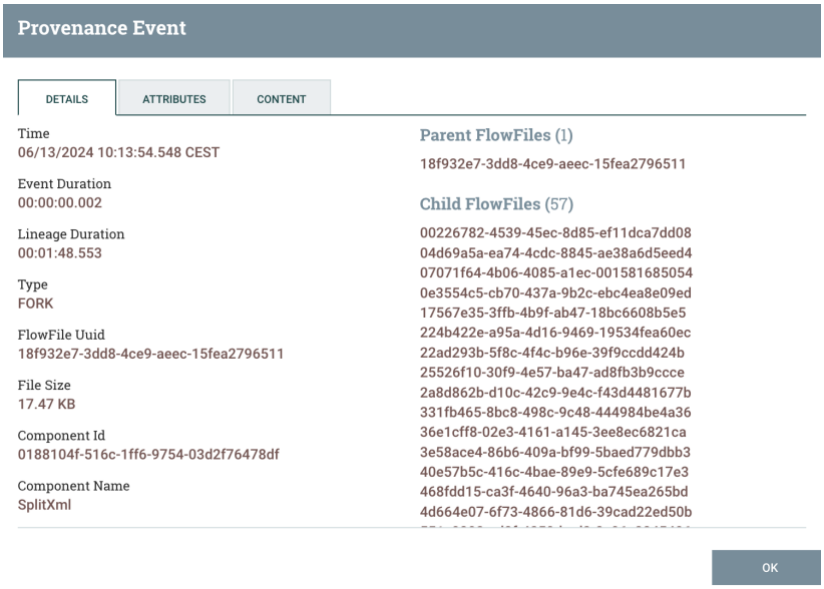
Captura de datos

En este anexo se adjuntan los datos capturados por Apache NiFi para con su formato de origen

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<amp:amp_folder_list version="1" xmlns:amp="http://www.airwave.com"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.airwave.com amp_folder_list.xsd">
<folder id="70">
<bandwidth_in>0</bandwidth_in>
<bandwidth_out>0</bandwidth_out>
<client_count>0</client_count>
<down>0</down>
<mismatch>0</mismatch>
<name>OVD.Rectorado</name>
<parent_id>24</parent_id>
<up>3</up>
<vpn_client_count>0</vpn_client_count>
</folder>
<folder id="17">
…
<client_count>46</client_count>
<down>0</down>
<mismatch>0</mismatch>
<name>Profesorado Norte</name>
<parent_id>25</parent_id>
<up>22</up>
<vpn_client_count>0</vpn_client_count>
</folder>
<folder id="43">
<bandwidth_in>6339177</bandwidth_in>
<bandwidth_out>33133964</bandwidth_out>
<client_count>131</client_count>
<down>0</down>
<mismatch>0</mismatch>
<name>Departamental Oeste</name>
<parent_id>18</parent_id>
<up>66</up>
<vpn_client_count>0</vpn_client_count>
</folder>
<folder id="51">
<bandwidth_in>0</bandwidth_in>
<bandwidth_out>0</bandwidth_out>
<client_count>5</client_count>
<down>0</down>
<mismatch>0</mismatch>
<name>Ed. Principado (Verde)</name>
<parent_id>24</parent_id>
```

Transformación de datos

En la siguiente imagen se muestra una captura del proceso siguiente a la recolección, los datos son divididos por hijos.



En la siguiente imagen se muestra el resultado de uno de los hijos resultantes del proceso de Split. El id = 70 indica que se refiere al folder 70.

```
<?xml version="1.0" encoding="UTF-8"?><folder id="70"
xmlns:amp="http://www.airwave.com" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
    <bandwidth_in>0</bandwidth_in>
    <bandwidth_out>0</bandwidth_out>
    <client_count>0</client_count>
    <down>0</down>
    <mismatch>0</mismatch>
    <name>OVD.Rectorado</name>
    <parent_id>24</parent_id>
    <up>3</up>
    <vpn_client_count>0</vpn_client_count>
  </folder>
```

En las siguientes imágenes se ponen los metadatos necesarios para cumplir con el esquema de validación.

```
<json:map xmlns:json="http://www.w3.org/2005/xpath-functions">
   <json:string key="organizationid">Universidad de Oviedo</json:string>
   <json:string key="typemeter">INAL</json:string>
   <json:number key="timestamp">1718266500183</json:number>
   <json:number key="timestamp_to">1718266500183</json:number>
   <json:number key="lat">43.3611</json:number>
   <json:number key="lon">-5.846443</json:number>
   <json:number key="cota">0</json:number>
   <json:string key="timezone">Europe/Madrid</json:string>
   <json:string key="description_origin">24 &gt; 70</json:string>
   <json:string key="origin">OVD.Rectorado</json:string>
   <json:string key="uid">70.OVD.Rectorado</json:string>
   <json:string key="alias">OVD.Rectorado</json:string>
   <json:array key="data">
      <json:map>
         <json:string key="name">Node connections</json:string>
         <json:number key="value">0</json:number>
         <json:string key="metric">connections</json:string>
         <json:string key="description">Total number of users associated to the device
regardless of which radio they are associated to, at the time of the last
polling.</json:string>
      </json:map>
      <json:map>
         <json:string key="name">Node vpn_connections</json:string>
         <json:number key="value">0</json:number>
         <json:string key="metric">connections</json:string>
         <json:string key="description">Number of VPN connections</json:string>
      </json:map>
      <json:map>
         <json:string key="name">Node bandwidth_in</json:string>
         <json:number key="value">0</json:number>
         <json:string key="metric">bandwidth</json:string>
         <json:string key="description">In bandwidth</json:string>
      </json:map>
      <json:map>
         <json:string key="name">Node bandwidth_out</json:string>
         <json:number key="value">0</json:number>
         <json:string key="metric">bandwidth</json:string>
         <json:string key="description">Out bandwidth</json:string>
      </json:map>
      <json:map>
         <json:string key="name">Nodes up</json:string>
         <json:number key="value">3</json:number>
         <json:string key="metric">nodes</json:string>
         <json:string key="description">Up nodes</json:string>
      </json:map>
      <json:map>
         <json:string key="name">Nodes down</json:string>
         <json:number key="value">0</json:number>
         <json:string key="metric">nodes</json:string>
         <json:string key="description">Down nodes</json:string>
      </json:map>
      <json:map>
         <json:string key="name">Nodes mismatch</json:string>
         <json:number key="value">0</json:number>
         <json:string key="metric">nodes</json:string>
         <json:string key="description">Mismatch nodes</json:string>
      </json:map>
   </json:array>
</json:map>
```

```
<json:map xmlns:json="http://www.w3.org/2005/xpath-functions">
   <json:string key="organizationid">Universidad de Oviedo</json:string>
   <json:string key="typemeter">INAL</json:string>
   <json:number key="timestamp">1718266500183</json:number>
   <json:number key="timestamp_to">1718266500183</json:number>
   <json:number key="lat">43.3611</json:number>
   <json:number key="lon">-5.846443</json:number>
   <json:number key="cota">0</json:number>
   <json:string key="timezone">Europe/Madrid</json:string>
   <json:string key="description_origin">24 &gt; 70</json:string>
   <json:string key="origin">OVD.Rectorado</json:string>
   <json:string key="uid">70.OVD.Rectorado</json:string>
   <json:string key="alias">OVD.Rectorado</json:string>
   <json:array key="data">
      <json:map>
         <json:string key="name">Node connections</json:string>
         <json:number key="value">0</json:number>
         <json:string key="metric">connections</json:string>
         <json:string key="description">Total number of users associated to the device
regardless of which radio they are associated to, at the time of the last
polling.</json:string>
      </json:map>
      <json:map>
         <json:string key="name">Node vpn_connections</json:string>
         <json:number key="value">0</json:number>
         <json:string key="metric">connections</json:string>
         <json:string key="description">Number of VPN connections</json:string>
      </json:map>
      <json:map>
         <json:string key="name">Node bandwidth_in</json:string>
         <json:number key="value">0</json:number>
         <json:string key="metric">bandwidth</json:string>
         <json:string key="description">In bandwidth</json:string>
      </json:map>
      <json:map>
         <json:string key="name">Node bandwidth_out</json:string>
         <json:number key="value">0</json:number>
         <json:string key="metric">bandwidth</json:string>
         <json:string key="description">Out bandwidth</json:string>
      </json:map>
      <json:map>
         <json:string key="name">Nodes up</json:string>
         <json:number key="value">3</json:number>
         <json:string key="metric">nodes</json:string>
         <json:string key="description">Up nodes</json:string>
      </json:map>
      <json:map>
         <json:string key="name">Nodes down</json:string>
         <json:number key="value">0</json:number>
         <json:string key="metric">nodes</json:string>
         <json:string key="description">Down nodes</json:string>
      </json:map>
      <json:map>
         <json:string key="name">Nodes mismatch</json:string>
         <json:number key="value">0</json:number>
         <json:string key="metric">nodes</json:string>
         <json:string key="description">Mismatch nodes</json:string>
      </json:map>
   </json:array>
</json:map>
```

```
<json:map xmlns:json="http://www.w3.org/2005/xpath-functions">
   <json:string key="topic_name">uo.pruebas.raw</json:string>
   <json:array key="arrayobjects">
      <json:map>
         <json:string key="organizationid">Universidad de Oviedo</json:string>
         <json:string key="typemeter">INAL</json:string>
         <json:number key="timestamp">1718266500183</json:number>
         <json:number key="timestamp_to">1718266500183</json:number>
         <json:number key="lat">43.3611</json:number>
         <json:number key="lon">-5.846443</json:number>
         <json:number key="cota">0</json:number>
         <json:string key="timezone">Europe/Madrid</json:string>
         <json:string key="description_origin">24 &gt; 70</json:string>
         <json:string key="origin">OVD.Rectorado</json:string>
         <json:string key="uid">70.OVD.Rectorado</json:string>
         <json:string key="alias">OVD.Rectorado</json:string>
         <json:array key="data">
            <json:map>
               <json:string key="name">Node connections</json:string>
               <json:number key="value">0</json:number>
               <json:string key="metric">connections</json:string>
               <json:string key="description">Total number of users associated to the
device regardless of which radio they are associated to, at the time of the last
polling.</json:string>
            </json:map>
            <json:map>
               <json:string key="name">Node vpn_connections</json:string>
               <json:number key="value">0</json:number>
               <json:string key="metric">connections</json:string>
               <json:string key="description">Number of VPN connections</json:string>
            </json:map>
            <json:map>
               <json:string key="name">Node bandwidth_in</json:string>
               <json:number key="value">0</json:number>
               <json:string key="metric">bandwidth</json:string>
               <json:string key="description">In bandwidth</json:string>
            </json:map>
            <json:map>
               <json:string key="name">Node bandwidth_out</json:string>
               <json:number key="value">0</json:number>
               <json:string key="metric">bandwidth</json:string>
               <json:string key="description">Out bandwidth</json:string>
            </json:map>
            <json:map>
               <json:string key="name">Nodes up</json:string>
               <json:number key="value">3</json:number>
               <json:string key="metric">nodes</json:string>
               <json:string key="description">Up nodes</json:string>
            </json:map>
            <json:map>
               <json:string key="name">Nodes down</json:string>
               <json:number key="value">0</json:number>
               <json:string key="metric">nodes</json:string>
               <json:string key="description">Down nodes</json:string>
            </json:map>
            <json:map>
               <json:string key="name">Nodes mismatch</json:string>
               <json:number key="value">0</json:number>
               <json:string key="metric">nodes</json:string>
               <json:string key="description">Mismatch nodes</json:string>
            </json:map>
         </json:array>
      </json:map>
   </json:array>
</json:map>
```

Teniendo ya los datos transformados en la siguiente imagen se muestra el resultado de la transformación a JSON


```
{
"topic_name": "uo.pruebas.raw",
"arrayobjects": [
{
"organizationid": "Universidad de Oviedo",
"typemeter": "INAL",
"timestamp": 1.718266500183E12,
"timestamp_to": 1.718266500183E12,
"lat": 43.3611,
"lon": -5.846443,
"cota": 0,
"timezone": "Europe\/Madrid",
"description_origin": "24 &gt; 70",
"origin": "OVD.Rectorado",
"uid": "70.OVD.Rectorado",
"alias": "OVD.Rectorado",
"data": [
{
"name": "Node connections",
"value": 0,
"metric": "connections",
"description": "Total number of users associated to the device regardless of
which radio they are associated to, at the time of the last polling."
},
{
"name": "Node vpn_connections",
"value": 0,
"metric": "connections",
"description": "Number of VPN connections"
},
```


Valores Obtenidos de Dispositivos Inalámbricos LoRa

Captura de datos

En este anexo se muestran los datos capturados en Apache NiFi con el formato de origen.

```
{
"end_device_ids": {
"device_id": "eui-70b3d57ed005cc33",
"application_ids": {
"application_id": "cubecell-bmp280-uo265941"
},
"dev_eui": "70B3D57ED005CC33",
"join_eui": "0123456789101112",
"dev_addr": "260B3BB7"
},
"correlation_ids": [
"gs:uplink:01J15HCZH5NE9HE663HZ7QMKP2"
],
"received_at": "2024-06-24T16:31:23.892062870Z",
"uplink_message": {
"session_key_id": "AY/OBKh1mOK2wBHRbcAfDw==",
"f_port": 2,
"f_cnt": 66701,
"frm_payload": "CX4AZAABjgUObA==",
"decoded_payload": {
"battery": 3.692,
"humidity": 100,
"pressure": 1018.93,
"temperature": 24.3
},
"rx_metadata": [
{
"gateway_ids": {
"gateway_id": "eui-3530362029003100",
"eui": "3530362029003100"
},
"time": "2024-06-24T16:29:41.726301Z",
"timestamp": 3074723656,
"rssi": -99,
"channel_rssi": -99,
"snr": 6.75,
"location": {
"latitude": 43.52408188947684,
"longitude": -5.634954927371218,
"source": "SOURCE_REGISTRY"
},
"uplink_token":
"CiIKIAoUZXVpLTM1MzAzNjIwMjkwMDMxMDASCDUwNiApADEAEMiekroLGgwI277mswYQv/m+xgIgwOK+nr6s
jAE=",
"channel_index": 4,
"received_at": "2024-06-24T16:31:23.658635219Z"
}
],
"settings": {
"data_rate": {
"lora": {
"bandwidth": 125000,
"spreading_factor": 7,
"coding_rate": "4/5"
}
},
"frequency": "867300000",
"timestamp": 3074723656,
"time": "2024-06-24T16:29:41.726301Z"
},
"received_at": "2024-06-24T16:31:23.687479871Z",
"consumed_airtime": "0.061696s",
"packet_error_rate": 0.04761905,
"locations": {
"user": {
```

Transformación

En este anexo se muestra las trazas de cada una de las transformaciones aplicadas al Flowfile sometido a pruebas para cumplir con el esquema estandarizado de almacenamiento


```
{
  "alias" : "cubecell-bmp280-uo265941",
  "origin" : "cubecell-bmp280-uo265941",
  "uid" : "eui-70b3d57ed005cc33",
  "data" : [ {
    "description" : "Battery voltage",
    "metric" : "v",
    "name" : "battery",
    "value" : 3.692
  }, {
    "metric" : "%",
    "description" : "Humidity percentage",
    "name" : "humidity",
    "value" : 100
  }, {
    "metric" : "hPa",
    "description" : "Pressure",
    "name" : "pressure",
    "value" : 1018.93
  }, {
    "metric" : "°C",
    "description" : "Temperature",
    "name" : "temperature",
    "value" : 24.3
  } ],
  "cota" : 5,
  "lat" : 43.52398440799855,
  "lon" : -5.635266396697048,
  "organizationid" : "Universidad de Oviedo",
  "timestamp" : 1719246683000,
  "timezone" : "Europe/Madrid",
  "timestamp_to" : 1719246683000,
  "description_origin" : "band_id: ; brand_id: ; firmware_version: ;
hardware_version: ; model_id: ",
  "typemeter" : "MQTT"
}
```

```
{
    "topic_name": "uo.pruebasdev.raw",
    "arrayobjects": [
        {
            "alias": "cubecell-bmp280-uo265941",
            "origin": "cubecell-bmp280-uo265941",
            "uid": "eui-70b3d57ed005cc33",
            "data": [
                {
                    "description": "Battery voltage",
                    "metric": "v",
                    "name": "battery",
                    "value": 3.692
                },
                {
                    "metric": "%",
                    "description": "Humidity percentage",
                    "name": "humidity",
                    "value": 100
                },
                {
                    "metric": "hPa",
                    "description": "Pressure",
                    "name": "pressure",
                    "value": 1018.93
                },
                {
                    "metric": "°C",
                    "description": "Temperature",
                    "name": "temperature",
                    "value": 24.3
                }
            ],
            "cota": 5,
            "lat": 43.52398440799855,
            "lon": -5.635266396697048,
            "organizationid": "Universidad de Oviedo",
            "timestamp": 1719246683000,
            "timezone": "Europe/Madrid",
            "timestamp_to": 1719246683000,
            "description_origin": "band_id: ; brand_id: ; firmware_version: ; hardware_version: ;
model_id: ",
            "typemeter": "MQTT"
        }
    ]
}
```

```
{
    "topic_name": "uo.pruebasdev.raw",
    "arrayobjects": [
        {
            "alias": "cubecell-bmp280-uo265941",
            "origin": "cubecell-bmp280-uo265941",
            "uid": "eui-70b3d57ed005cc33",
            "data": [
                {
                    "description": "Battery voltage",
                    "metric": "v",
                    "name": "battery",
                    "value": 3.692
                },
                {
                    "metric": "%",
                    "description": "Humidity percentage",
                    "name": "humidity",
                    "value": 100
                },
                {
                    "metric": "hPa",
                    "description": "Pressure",
                    "name": "pressure",
                    "value": 1018.93
                },
                {
                    "metric": "°C",
                    "description": "Temperature",
                    "name": "temperature",
                    "value": 24.3
                }
            ],
            "cota": 5,
            "lat": 43.52398440799855,
            "lon": -5.635266396697048,
            "organizationid": "Universidad de Oviedo",
            "timestamp": 1719246683000,
            "timezone": "Europe/Madrid",
            "timestamp_to": 1719246683000,
            "description_origin": "band_id: ; brand_id: ; firmware_version: ; hardware_version: ;
model_id: ",
            "typemeter": "MQTT"
        }
    ]
}
```

Prueba de rendimiento

En la Ilustración 57 se muestra la carga normalizada del sistema, que es una medida de la carga de trabajo relativa en comparación con la capacidad total del sistema. Una carga normalizada mayor indica una mayor demanda de recursos en relación con la capacidad disponible.

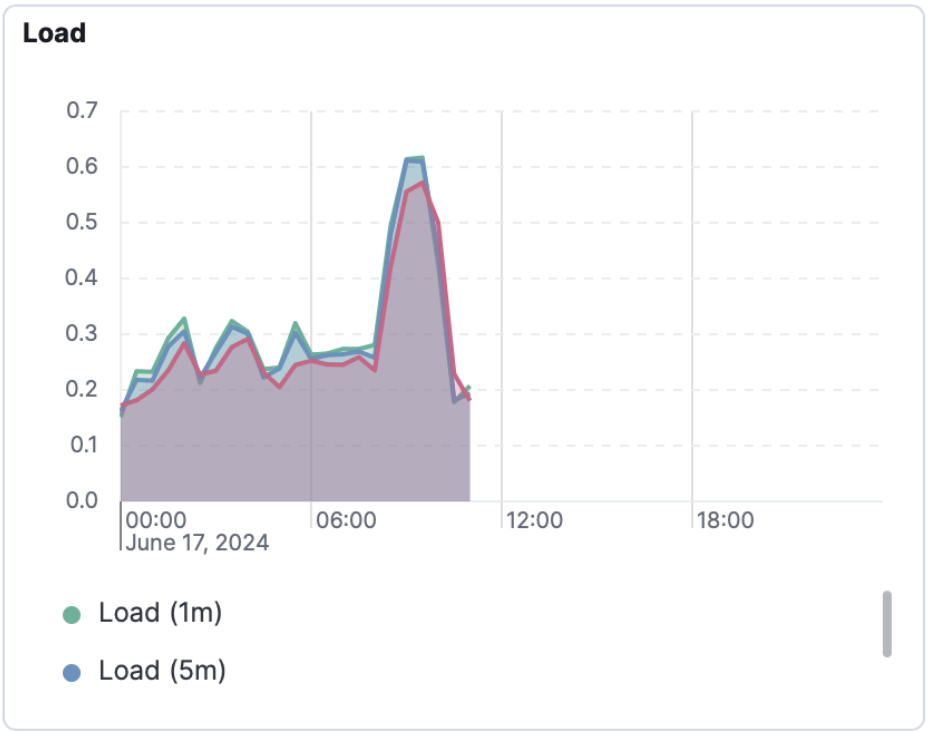


Ilustración 57. Captura de MetricBeat correspondiente a prueba de rendimiento. Load (Carga Normalizada)

En la Ilustración 58 se muestra el uso del disco en el clúster. Muestra cómo se utiliza el espacio de almacenamiento del disco durante la prueba de carga. Un aumento en el uso de disco podría indicar una mayor escritura o lectura de datos durante la carga intensiva.

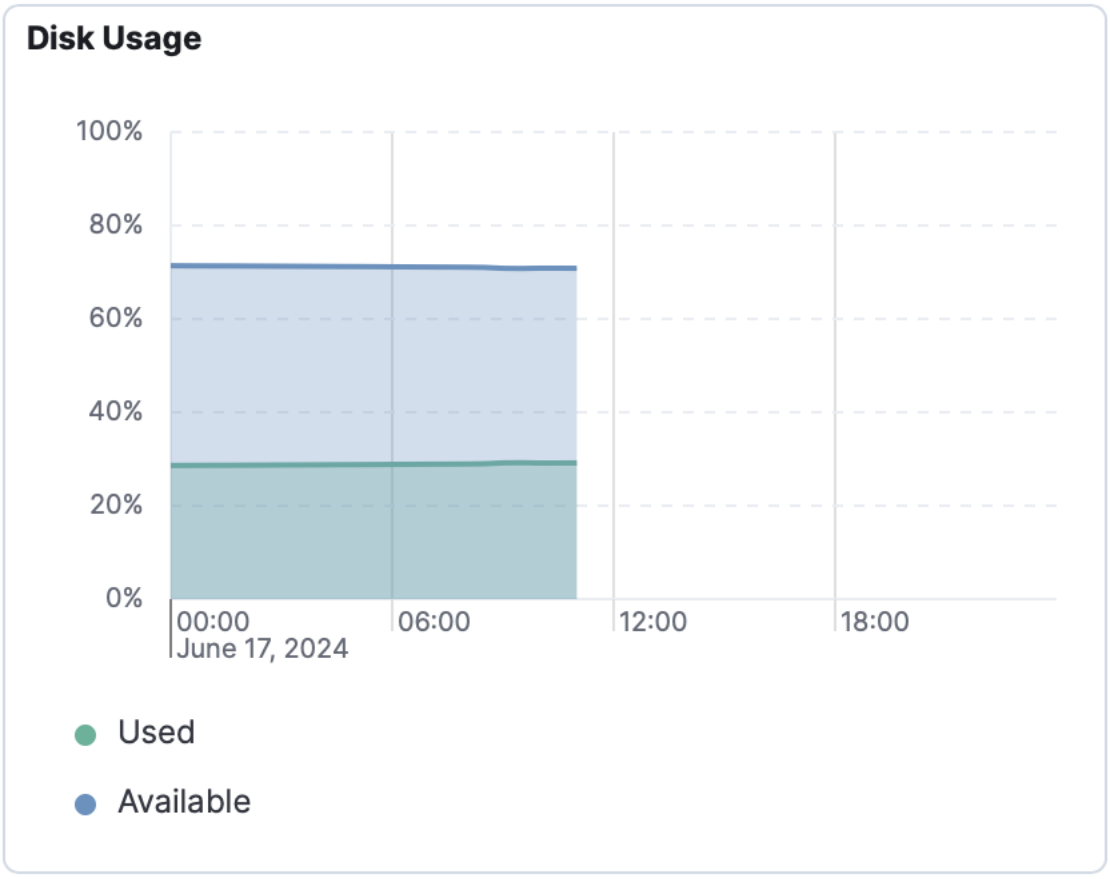


Ilustración 58. Captura de MetricBeat correspondiente a prueba de rendimiento. Disk Usage (Uso de Disco)

En la Ilustración 59 se representa el uso de memoria en el clúster. Indica cómo se está utilizando la memoria RAM disponible durante la ejecución de la prueba de rendimiento. Un aumento en el uso de memoria podría implicar una mayor demanda de procesamiento de datos en la memoria.

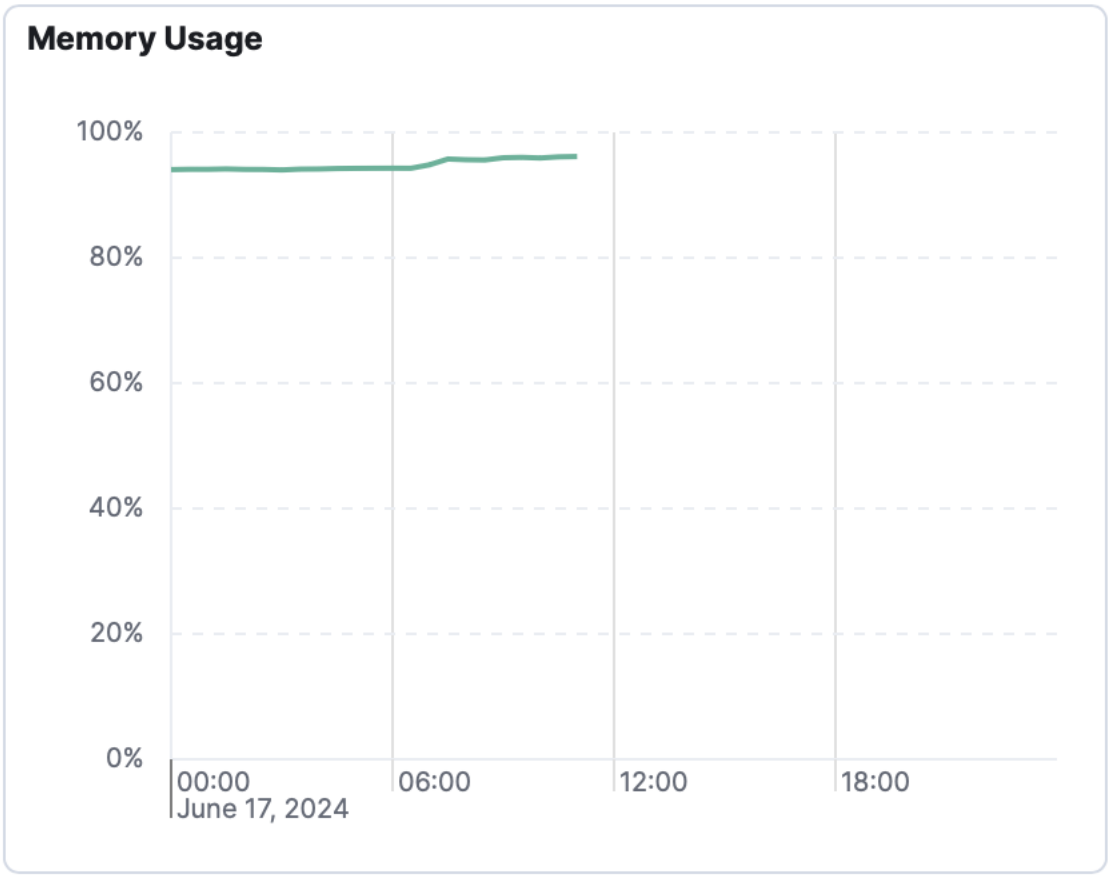


Ilustración 59. Captura de MetricBeat correspondiente a prueba de rendimiento. Memory Usage (Uso de Memoria)

En la Ilustración 60 se muestra el uso de la CPU del sistema. Indica el porcentaje de capacidad de procesamiento de la CPU que se está utilizando durante la prueba. Un aumento en el uso de CPU podría reflejar una mayor carga de trabajo computacional durante la prueba.

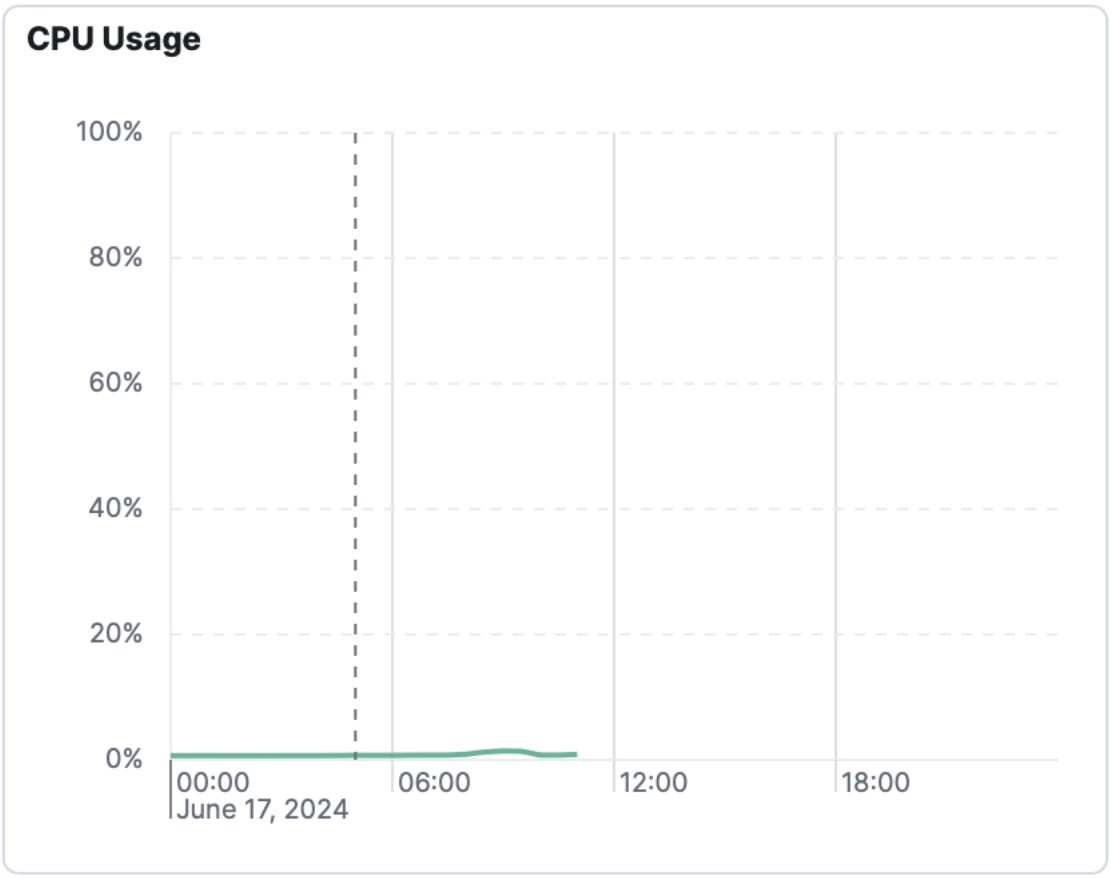


Ilustración 60. Captura de MetricBeat correspondiente a prueba de rendimiento .CPU Usage (Uso de CPU):

ABREVIATURAS

- MQTT: Message Queuing Telemetry Transport
- IoT: Internet of Things
- ETL: Extract, transform, and load (Extracción, Transformación y Carga)
- TTN: The Things Network
- ELK: Elasticsearch, Logstash, Kibana
- DBMS: Data Base Management System
- SGBD: Sistema de Gestión de Bases de Datos
- SIUU: Sistema Integrado de Información Universitaria
- LoRaWAN: Long Range Wide Area Network
- OPC UA: Open Platform Communications Unified Architecture
- HTTP REST: Hypertext Transfer Protocol Representational State Transfer
- SCK : Smart Citizen Kit
- JSON: JavaScript Object Notation
- XML: Extensible Markup Language
- CSV: Comma-Separated Values
- MPP: Analytical Massively Parallel Processing .Procesamiento Masivamente Paralelo
- TI: Tecnología de la información o IT(Information Technology).
- RFID: Radio Frequency Identification
- API : Application Programming Interface
- SIEM: Security Information and Event Management
- SOC: Centro de Operaciones de Seguridad
- UPM: Universidad Politécnica de Madrid
- NSA: Agencia de Seguridad Nacional de los Estados Unidos
- JVM: Java Virtual Machine
- SaaS: El software como servicio
- DSL: Domain Specific Language
- WBS: Work Breakdown Structure

REFERENCIAS BIBLIOGRÁFICAS

1. Matacuta A, Popa C. Big Data Analytics: Analysis of Features and Performance of Big Data Ingestion Tools. Inform Econ. 30 de junio de 2018;22(2/2018):25-34.

2. Isah H, Zulkernine F. A Scalable and Robust Framework for Data Stream Ingestion. En: 2018 IEEE International Conference on Big Data (Big Data) [Internet]. Seattle, WA, USA: IEEE; 2018 [citado 19 de abril de 2024]. p. 2900-5. Disponible en: https://ieeexplore.ieee.org/document/8622360/

3. Grünewald I. Experiencia de la Universidad del Bío-Bío en el desarrollo de un sistema de gestión integrado en información universitaria. Calid En Educ. 7 de mayo de 2006;(24):235-46.

4. Regueiro MA, Viqueira JRR, Taboada JA, Cotos JM. Virtual integration of sensor observation data. Comput Geosci. agosto de 2015;81:12-9.

5. Gil S, Zapata-Madrigal GD, García-Sierra R, Cruz Salazar LA. Converging IoT protocols for the data integration of automation systems in the electrical industry. J Electr Syst Inf Technol. diciembre de 2022;9(1):1.

6. Chhaya L, Sharma P, Kumar A, Bhagwatikar G. IoT-Based Implementation of Field Area Network Using Smart Grid Communication Infrastructure. Smart Cities. 14 de diciembre de 2018;1(1):176-89.

7. Alvarez-Campana M, López G, Vázquez E, Villagrá V, Berrocal J. Smart CEI Moncloa: An IoT-based Platform for People Flow and Environmental Monitoring on a Smart University Campus. Sensors. 8 de diciembre de 2017;17(12):2856.

8. Sutjarittham T, Gharakheili HH, Kanhere SS, Sivaraman V. Realizing a Smart University Campus: Vision, Architecture, and Implementation. En: 2018 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS) [Internet]. Indore, India: IEEE; 2018 [citado 19 de febrero de 2024]. p. 1-6. Disponible en: https://ieeexplore.ieee.org/document/8710084/

9. Maciá Pérez F, Berna Martínez JV, Lorenzo Fonseca I. Modelling and Implementing Smart Universities: An IT Conceptual Framework. Sustainability. 19 de marzo de 2021;13(6):3397.

10. Cheng Y, Wu Z. Design of Digital Campus Platform Data Integration System Based on Xml. En: 2019 International Conference on Intelligent Transportation, Big Data & Smart

City (ICITBS) [Internet]. Changsha, China: IEEE; 2019 [citado 3 de junio de 2024]. p. 253-8. Disponible en: https://ieeexplore.ieee.org/document/8669640/

11. Bo L, Shi-Hai Y, Yong-Biao Y. Intelligent Integration Method of Big Data for Time Sharing Regulation of Elastic Load. En: 2020 IEEE International Conference on Industrial Application of Artificial Intelligence (IAAI) [Internet]. Harbin, China: IEEE; 2020 [citado 3 de junio de 2024]. p. 266-71. Disponible en: https://ieeexplore.ieee.org/document/9332875/

12. Sivabalan S, Minu RI. Heterogeneous Data Integration with ELT and Analytical MPP Database for Data Analysis Application. En: 2021 Innovations in Power and Advanced Computing Technologies (i-PACT) [Internet]. Kuala Lumpur, Malaysia: IEEE; 2021 [citado 3 de junio de 2024]. p. 1-5. Disponible en: https://ieeexplore.ieee.org/document/9696841/

13. Wnęk K, Boryło P. A Data Processing and Distribution System Based on Apache NiFi. Photonics. 15 de febrero de 2023;10(2):210.

14. Liu J, Braun E, Dupmeier C, Kuckertz P, Ryberg DS, Robinius M, et al. A Generic and Highly Scalable Framework for the Automation and Execution of Scientific Data Processing and Simulation Workflows. En: 2018 IEEE International Conference on Software Architecture (ICSA) [Internet]. Seattle, WA: IEEE; 2018 [citado 17 de abril de 2024]. p. 145-14510. Disponible en: https://ieeexplore.ieee.org/document/8417148/

15. Dhaouadi A, Paccoud W, Bousselmi K, Monnet S, Gammoudi MM, Hammoudi S. Big Data Tools: Interoperability Study and Performance Testing. En: 2023 IEEE International Conference on Big Data (BigData) [Internet]. Sorrento, Italy: IEEE; 2023 [citado 17 de abril de 2024]. p. 2386-95. Disponible en: https://ieeexplore.ieee.org/document/10386089/

16. Alwidian J, Rahman SA, Gnaim M, Al-Taharwah F. Big Data Ingestion and Preparation Tools. Mod Appl Sci. 27 de agosto de 2020;14(9):12.

17. Irfan M, Reena, George J. Data Ingestion - Cloud based Ingestion Analysis using NiFi. En: 2023 International Conference on Self Sustainable Artificial Intelligence Systems (ICSSAS) [Internet]. Erode, India: IEEE; 2023 [citado 18 de abril de 2024]. p. 1-9. Disponible en: https://ieeexplore.ieee.org/document/10331884/

18. Carthen C, Zaremehrjardi A, Le V, Cardillo C, Strachan S, Tavakkoli A, et al. Orchestrating Apache NiFi/MiNiFi within a Spatial Data Pipeline. En: 2023 IEEE/ACIS 21st International Conference on Software Engineering Research, Management and Applications (SERA) [Internet]. Orlando, FL, USA: IEEE; 2023 [citado 18 de abril de 2024]. p. 366-71. Disponible en: https://ieeexplore.ieee.org/document/10197731/

19. Racka K. APACHE NIFI AS A TOOL FOR STREAM PROCESSING OF MEASUREMENT DATA. Nauki Ekon [Internet]. 3 de agosto de 2022 [citado 18 de abril de

2024];(TOM XXXV). Disponible en: https://doi.org/10.19251/ne/2022.35(7)

20. Chanthakit S, Keeratiwintakorn P, Rattanapoka C. An IoT System Design with Real-Time Stream Processing and Data Flow Integration. En: 2019 Research, Invention, and Innovation Congress (RI2C) [Internet]. Bangkok, Thailand: IEEE; 2019 [citado 18 de abril de 2024]. p. 1-5. Disponible en: https://ieeexplore.ieee.org/document/8999968/

21. Kammachi HJ, H VP. Accelerating Data Migration: A Comparative Study of Apache Spark and Other Leading Tools. En: 2023 IEEE 11th Region 10 Humanitarian Technology Conference (R10-HTC) [Internet]. Rajkot, India: IEEE; 2023 [citado 18 de abril de 2024]. p. 790-4. Disponible en: https://ieeexplore.ieee.org/document/10461908/

22. Hlaing NN, Soe Nyunt TT. Developing Scalable and Lightweight Data Stream Ingestion Framework for Stream Processing. En: 2023 IEEE Conference on Computer Applications (ICCA) [Internet]. Yangon, Myanmar: IEEE; 2023 [citado 19 de abril de 2024]. p. 405-10. Disponible en: https://ieeexplore.ieee.org/document/10181764/

23. Bajer M. Building an IoT Data Hub with Elasticsearch, Logstash and Kibana. En: 2017 5th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW) [Internet]. Prague: IEEE; 2017 [citado 19 de febrero de 2024]. p. 63-8. Disponible en: http://ieeexplore.ieee.org/document/8113772/

24. Bhatnagar D, SubaLakshmi RJ, C V. Twitter Sentiment Analysis Using Elasticsearch, LOGSTASH And KIBANA. En: 2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE) [Internet]. Vellore, India: IEEE; 2020 [citado 22 de abril de 2024]. p. 1-5. Disponible en: https://ieeexplore.ieee.org/document/9077689/

25. Petrova-Antonova D, Baychev S, Pavlova I, Pavlov G. Air Quality Visual Analytics with Kibana. En: 2020 5th International Conference on Smart and Sustainable Technologies (SpliTech) [Internet]. Split, Croatia: IEEE; 2020 [citado 22 de abril de 2024]. p. 1-6. Disponible en: https://ieeexplore.ieee.org/document/9243708/

26. Ahmed F, Jahangir U, Rahim H, Ali K, Agha D e S. Centralized Log Management Using Elasticsearch, Logstash and Kibana. En: 2020 International Conference on Information Science and Communication Technology (ICISCT) [Internet]. Karachi, Pakistan: IEEE; 2020 [citado 22 de abril de 2024]. p. 1-7. Disponible en: https://ieeexplore.ieee.org/document/9080053/

27. Sankar P, George DE, S ASN. Social media monitoring using ELK Stack. En: 2022 IEEE International Conference on Signal Processing, Informatics, Communication and Energy Systems (SPICES) [Internet]. THIRUVANANTHAPURAM, India: IEEE; 2022 [citado

22 de abril de 2024]. p. 231-5. Disponible en: https://ieeexplore.ieee.org/document/9774273/

28. Son SJ, Kwon Y. Performance of ELK stack and commercial system in security log analysis. En: 2017 IEEE 13th Malaysia International Conference on Communications (MICC) [Internet]. Johor Bahru: IEEE; 2017 [citado 22 de abril de 2024]. p. 187-90. Disponible en: http://ieeexplore.ieee.org/document/8311756/

29. Gaonkar P, Lonkar A, Sasirekha G, Bapat J, Das D. Comfort Management System for Smart Buildings: An Open-Source Scalable Prototype. En: 2020 IEEE-HYDCON [Internet]. Hyderabad, India: IEEE; 2020 [citado 22 de abril de 2024]. p. 1-5. Disponible en: https://ieeexplore.ieee.org/document/9242704/

30. Kuduz N, Salapura S. Building a Multitenant Data Hub System using Elastic Stack and Kafka for Uniform Data Representation. En: 2020 19th International Symposium INFOTEH-JAHORINA (INFOTEH) [Internet]. East Sarajevo, Bosnia and Herzegovina: IEEE; 2020 [citado 22 de abril de 2024]. p. 1-6. Disponible en: https://ieeexplore.ieee.org/document/9066286/

31. Agrawal S, Sonbhadra SK, Agarwal S. Favour prediction of Taxi services using real-time visualization. En: 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI) [Internet]. Bangalore: IEEE; 2018 [citado 22 de abril de 2024]. p. 2276-82. Disponible en: https://ieeexplore.ieee.org/document/8554632/

32. Vega C, Roquero P, Leira R, Gonzalez I, Aracil J. Loginson: a transform and load system for very large-scale log analysis in large IT infrastructures. J Supercomput. septiembre de 2017;73(9):3879-900.

33. Scott-Boyer MP, Dufour P, Belleau F, Ongaro-Carcy R, Plessis C, Périn O, et al. Use of Elasticsearch-based business intelligence tools for integration and visualization of biological data. Brief Bioinform. 22 de septiembre de 2023;24(6):bbad348.

34. Satyanarayan A, Moritz D, Wongsuphasawat K, Heer J. Vega-Lite: A Grammar of Interactive Graphics. IEEE Trans Vis Comput Graph. enero de 2017;23(1):341-50.