# Diseño y aplicación de una arquitectura *IoT* para el Sistema de Aislamiento Limitado o Total (SAL/T)

Matías Sambrizzi† [1], Fernando Iglesias† [2], Dr. Ariel Lutenberg*[2]
† *Facultad de Ingeniería - Universidad de Buenos Aires (FI-UBA)*
* *CONICET -GICSAFe*
[1]msambrizzi@fi.uba.ar
[2]fjiglesias@fi.uba.ar

***Resumen*—En el presente artículo se presentan los avances del desarrollo de una arquitectura modular para la Central Operativa *SAL/T* basado en el paradigma de Internet de las Cosas (IoT). Este proyecto es el complemento en materia de *software* de un prototipo realizado para Trenes Argentinos, entidad que se ocupa de operar la red ferroviaria de la Argentina.**

***Palabras claves*—MQTT, JSON, ...**

## I. Motivación y Contexto

Las formaciones ferroviarias cuentan con diferentes sistemas de seguridad a bordo. Estos equipos se encargan de supervisar el correcto funcionamiento de los subsistemas críticos. Ante una falla en uno de los subsistemas, una formación ferroviaria se detiene inmediatamente por la activación automática de las señales de corte de tracción (*CT*) y frenado de emergencia (*FE*). En esta situación, el conductor debe llevar la formación a un lugar seguro para que los pasajeros puedan descender y, posteriormente, trasladarla a un taller para que pueda ser reparada.

El *SAL/T* [1] [1], según sus siglas, Sistema de Aislamiento Limitado o Total, es un sistema que le permite al maquinista de una formación ferroviaria la posibilidad de activar y desactivar el modo aislado limitado. En este modo, el equipo permite la circulación de la formación al desactivar las señales de corte de tracción y freno de emergencia generadas por los subsistemas críticos. Para que esta operación se realice de forma segura, se debe monitorear la velocidad de la formación tal que sea posible evitar que supere cierto valor máximo.

A partir del prototipo físico del SAL/T previamente mencionado, se presentan los avances en el desarrollo de una central operativa conformando una solución integral que posibilita la administación, la configuración y el monitoreo en tiempo real de la información recibida y transmitida por parte de cada dispositivo SAL/T desde una plataforma digital.

## II. Descripción de la problemática a resolver

Los subsistemas asociados al SAL/T como la seguridad de puertas, el sistema de hombre vivo y la protección de coche a la deriva; son críticos debido a que, en caso de fallar, pueden ocasionar lesiones o muertes de personas e incluso generar pérdidas materiales.

La central operativa permite la administación y configuración en forma remota de los dispositivos de supervisión de seguridad de cada formación ferroviaria, la visualización de los diferentes parámetros de interés involucrados tal que se encuentre al alcance de una o más personas asignadas dentro de una entidad y de este modo, sea posible optimizar la toma de decisiones.

Como consecuencia de lo expresado previamente, se ha realizado el diseño de una arquitectura versátil y confiable en términos de capacidad de procesamiento y de almacenamiento de la información.

En la sección III, según el *stack* [2] [2] de una solución IoT, se presentan cada una de las tecnologías utilizadas. Luego, en la sección IV se exponen las conclusiones y por último, en la sección V, se presentan los próximos pasos del trabajo.

## III. Arquitectura propuesta

En el marco del presente trabajo, la arquitectura esta estratificada en capas tal como se ilustra en la fiugra X. En las siguientes subsecciones se describen cada una de estas capas.

### A. Capa de dispositivos

Sobre la base de un prototipo del Sistema de Aislamiento Limitado o Total, *SAL/T*, se está desarrollando, en el kit de desarrollo *Nucleo F429*, un cliente *MQTT* que permite la publicación de la información proveniente de los subsistemas de falla, la velocidad de la información, entre otras variables; y llegado el caso, el dispositivo pueda realizar la lectura de diferentes parámetros configurables.

### B. Capa de conectividad

Para la comunicación entre los dispositivos *SAL/T* y la capa de almacenamiento, se utiliza un *Broker MQTT* [3] [3] que oficia de orquestador entre el cliente, dispositivo *SAL/T*, que publica los mensajes y aquel que se suscribe, el sistema *Kafka* [4] [4], para la lectura de la información y su posterior procesamiento.

[1]referencia al artículo de Di Vito

[2]agregar referencia
[3]agregar referencia
[4]agregar referencia

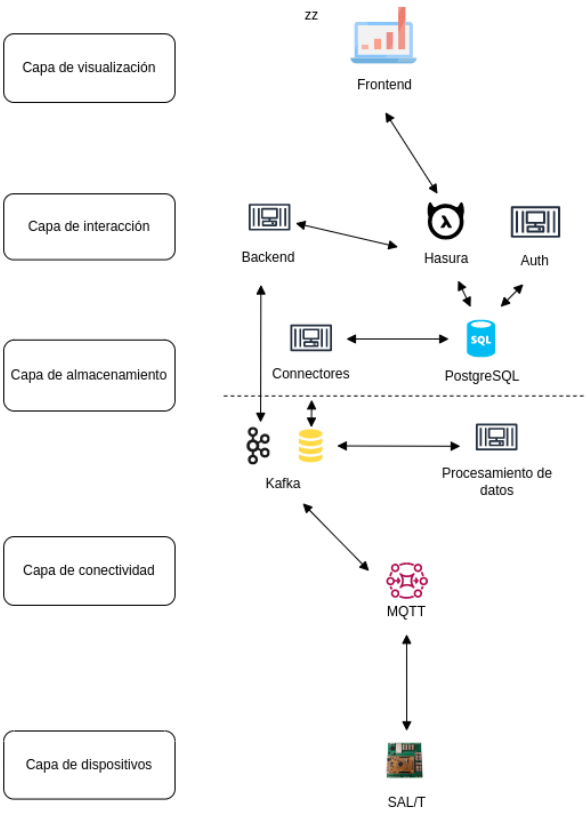


Fig. 1. Arquitectura de la Central Operativa SAL/T.

Dado que los mensajes intercambiados entre las partes contienen información sensible, se ha optado por agregar la capa de seguridad *TLS* [5] [5]. En este sentido, resulta indispensable utilizar el certificado *X509* [6] [6] para prevenir ataques del tipo *adversary-in-the-middle* [7] [7] y utilizar certificados emitidos por autoridades reconocidas.

### C. Capa de almacenamiento

La capa de almacenamiento por un lado se encuentra constituida por el sistema distribuido *Kafka* que se encarga de almacenar los datos de la aplicación. Entre ellos se destacan los mensajes MQTT enviados por los dispositivos SAL/T, como también la información referida a las formaciones y a los SAL/T que las ocupan.

Además, se tienen servicios de procesamiento de datos que se encargan de adaptar la información de los tópicos en datos que posteriormente serán consumidos por los servicios que integran la capa superior. Por otro lado, se tiene una base de datos *SQL* [8] [8] que se utiliza para almacenar los datos del servicio de autenticación y el motor que facilita la interacción con la plataforma web. Los conectores se encargan de insertar los datos que se quieran disponer a la capa de visualización.

[5] agregar referencia
[6] agregar referencia
[7] agregar referencia
[8] agregar referencia

### D. Capa de interacción

La capa de interacción se encuentra conformada por tres unidades. Por un lado, se ha desarrollado en el lenguaje *Kotlin* [9] [9] en conjunto con el *framework Ktor* [10] [10] un microservicio de autenticación donde es posible la gestión de los usuarios con sus respectivos roles y también el *backend*. Entre sus tareas principales se encuentran las relacionadas al protocolo *MQTT* para la actualización de los certificados que brindan seguridad a las comunicaciones, mediante el uso de la capa de seguridad *TLS*, y la indexación de los eventos que se publiquen en tiempo real; como la configuración general que permite el funcionamiento integral de los servicios y módulos dispuestos en el sistema.

Además, se dispone del motor *Hasura* [11] [11], el cual se encuentra conectado a una base de datos relacional (*PostgreSQL* [12] [12]) basado en el lenguaje de consulta estructurado *SQL*, que permite exponer una *API GraphQL* [13] [13] a aquellos clientes que deseen obtener una visualización del modelo de datos propuesto en que se conjuga la información de cada formación ferroviar con su correspondiente dispositivo *SAL/T* y diseño destinado al *frontend*. Más aún, el enlace permite realizar modificaciones y hasta remociones de las columnas propuestas en las tablas de cada esquema dentro de la base de datos.

### E. Capa de visualizacion

En esta capa, se ha utilizato el lenguaje de programación *TypeScript* [14] [14] junto con la biblioteca gráfica *React* [15] [15] para brindar una web reactiva en el que se elaboraron los formularios, las tablas y los paneles a partir de la explotación en de los datos almacenados en la base de datos. Cabe destacar que la plataforma web se encuentra condicionada por el rol y la entidad a la que pertenece cada usuario.

AGREGAR IMAGEN/ES DE LA PLATAFORMA

## IV. Conclusiones

En este trabajo se presentan las principales características del diseño de una arquitectura en materia de *software* y de *firmware* para la operabilidad de los dispositivos SAL/T.

Entre los módulos desarrollados, se posibilita la visualización de los usuarios y los dispositivos de cada entidad ofreciendo una solución sencilla y escalable. También, se cuenta la ingestión de la información recibida desde los dispositivos permitiendo una alta disponibilidad y operabilidad de los recursos. Por último, se cuenta con un microservicio destinado a la gestión de usuarios y perfiles que posee una alta tolerancia a fallos y adaptable.

[9] agregar referencia
[10] agregar referencia
[11] agregar referencia
[12] agregar referencia
[13] agregar referencia
[14] agregar referencia
[15] agregar referencia

En la actualidad, se encuentra en desarrollo propiciar al operario enviar comandos de control a los dispositivos activos que tenga asignados, brindar la connfiguración de los parámetros de cada dispositivo y de ser deseable su correspondiente modificación y finalmente, brindar la seguridad en las comunicaciones agregando la capa *TLS* en el protocolo *MQTT*.

## REFERENCECIAS


[1] Tracking by detection, OpenCV, https://opencv.org/multiple-object-tracking-in-realtime/
[2] Object detection and Tracking, NVIDIA, https://www.nvidia.com/en-us/on-demand/session/gtcwashingtondc2018-dc8127/
[3] Retail Analytics, Intel, https://www.intel.com/content/www/us/en/retail/analytics/overview.html
[4] Object Detection, Amazon, https://docs.aws.amazon.com/sagemaker/latest/dg/object-detection.html
[5] Track objects, GCP, https://cloud.google.com/video-intelligence/docs/object-tracking?hl=es-419
[6] Jason Brownlee. What is computer vision?, machinelearningmastery.
[7] Jason Brownlee. What is Deep Learning?, machinelearningmastery.com
[8] Object Detection, Tensorflow, https://www.tensorflow.org/lite/examples/object%5Fdetection/overview
[9] Bounding box, OpenCV,https://www.delftstack.com/howto/python/opencv-bounding-box/
[10] Joseph Redmon y Ali Farhadi. YOLO: AN Incremental Improvement.
[11] Object Tracking, VISO AI, https://viso.ai/deep-learning/object-tracking/
[12] Dietrich Paulus Nicolai Wojke Alex Bewley. DeepSORT, Mar. 2020.
[13] Yongxin Yang Kaiyang Zhou. Omni-Scale Feature Learning for Person, Re-Identification. Dic. 2019.
[14] Active Control of Camera Parameters for Object Detection Algorithms, Yulong Wu, John Tsotsos, https://arxiv.org/pdf/1705.05685.pdf
[15] Scikit learn, scikit-learn.org, metrics, pairwise cosine similarity.
[16] Análisis Cluster, Universidad de Valencia, https://www.uv.es/ceaces/multivari/cluster/CLUSTER2.htm
[17] Electronic Arts. Los Sims 3.
[18] NVIDIA SDK, https://developer.nvidia.com/nvidia-sdk-manager
[19] NVIDIA Jetson, https://www.nvidia.com/es-la/autonomous-machines/embedded-systems/
[20] Frame Rate Matters for Embedded, Ashton Fagg, https://eprints.qut.edu.au/117072/1/Ashton%5FFagg%5FThesis.pdf
[21] Using Quantization with NVIDIA, NVIDIA Developer, https://developer.nvidia.com/blog/achieving-fp32-accuracy-for-int8-inference-using-quantization-aware-training-with-tensorrt/
[22] Globant S.A, https://www.globant.com/es/about