Diseño e implementación de una central operativa para el control y monitoreo en el material rodante

xxxxxx xxxxxxxxx, xxxxxxxx xxxxxxxx, xxxxx xxxxxxxxx
xxxxxxx xxxxxxxx
xxxxxxxxxxx xx xxxxxxxx xxxxxxxxx
xxxxxxxx xx xxxxxxxxxx - xxx
xxxxxx xxxxx - xxxxxxxxx [1]xxxxxxxxxxxxx

***Resumen*— Ante la falla de sistemas tales como el control de puertas o la cámara frontal de registro de accidentes viales las formaciones ferroviarias activan automáticamente el corte de tracción y el freno de emergencia. En esas circunstancias la formación queda detenida y los pasajeros deben descender a las vías si la detención no se produce en una estación ferroviaria. En este trabajo se presenta el desarrollo para Trenes Argentinos de una arquitectura modular basada en el paradigma de Internet de las Cosas (IoT) que permite visualizar y gestionar los sistemas involucrados en esas situaciones.**

***Palabras claves*—Sistemas ferroviarios, MQTT/TLS, TypeScript, Apache Kafka**

I. Motivación y Contexto

El sistema ferroviario de la República Argentina cuenta con una gran cantidad de formaciones ferroviarias en las que se encuentran 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], según sus siglas, Sistema de Aislamiento Limitado o Total, es un dispositivo del cual se cuenta con una primera versión prototipada; que se presenta como solución a las contingencias descriptas anteriormente. De esta manera, el maquinista de una formación ferroviaria cuenta con la posibilidad de activar y desactivar el modo ailsado limitado.

En el modo asilado limitado, 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 complete de forma segura, se debe monitorear la velocidad de la formación tal que sea posible evitar que supere cierto valor máximo.

A partir del desarrollo previo del prototipo, se presentan los avances en la implementación de una central operativa que centraliza los dispositivos SAL/T para su respectiva administación, configuración y monitoreo en tiempo real de la información recibida y transmitida desde una plataforma digital. El proyecto se desarrolla por xxxxxxx-xxxxxxx [2] para la empresa Trenes Argentinos [3].

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 administración 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 por las personas asignadas dentro de una entidad y de este modo, sea posible optimizar la toma de decisiones.

A partir de los modelos más relevantes que se encuentran establecidos para el desarrollo e implementación de aplicaciones de software se consideran el *stack web* y el *stack IoT* [4]. Dadas las similitudes entre las capas de ambos esquemas, excepto en la capa de aplicación, donde el stack web emplea el protocolo *HTTP* [5] mientras que el *stack IoT* utiliza el protocolo *MQTT* [6]; siendo este último el más adecuado para escenarios donde son limitados los recursos como el ancho de banda y el consumo de energía.

En especial, el protocolo *MQTT* dispone de mensajes más livianos y además es posible transmitir y/o recibir datos en formato binario sin la necesidad de una codificación previa. También, este protocolo permite asignar niveles de calidad de servicio (*QoS*) [7] a los mensajes transmitidos, resultando una característica primordial en aplicaciones donde la probabilidad de pérdidas de paquetes es considerable.

La sección III presenta cada una de las capas con sus respectivas tecnologías en base al stack descripto anteriormente. Luego, en la sección IV se exponen las conclusiones del trabajo.

III. Arquitectura propuesta

En el marco del presente trabajo, la arquitectura esta estratificada en capas tal como se ilustra en la figura 1. 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 desarolló, en el kit de desarrollo *Nucleo F429* [8], un cliente *MQTT* que permite la publicación de la velocidad de la formación y de los parámetros provistos por los subsistemas de falla, entre otras variables; y llegado

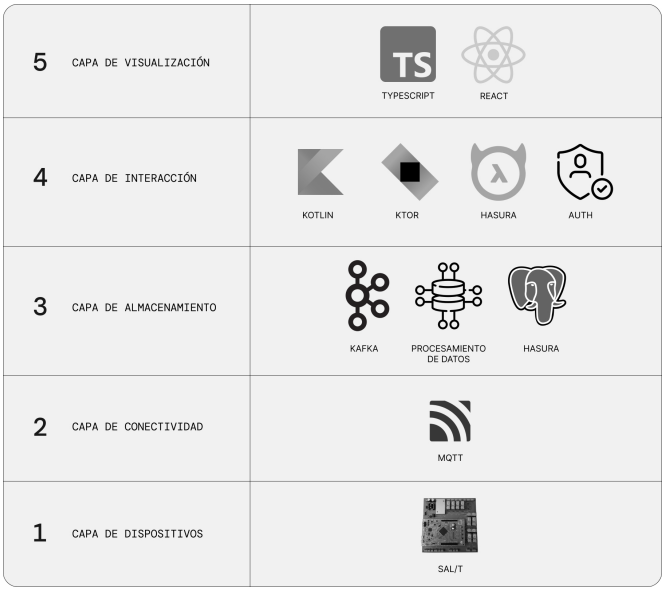

Fig. 1: Arquitectura de la Central Operativa SAL/T.

DISPOSITIVOS

id	train_id	status	subsystems	train
b8ce177e-9aa8-4606-bb83-75a58cf3c396	897ba9f3-35dd-4150-94c2-523c3919f164	disabled	Close	Close

TRENES

id	entity_id	series_number	entity
897ba9f3-35dd-4150-94c2-523c3919f164	67bc0bf0-9d1f-4986-a3da-897770f21cac	AXW123	Close

ENTIDADES

id	name	description
67bc0bf0-9d1f-4986-a3da-897770f21cac	BCYL	Belgrano Cargas

Fig. 2: Modelo de datos de los dispositivos, trenes y entidades.

DISPOSITIVOS

id	train_id	status	subsystems	train
b8ce177e-9aa8-4606-bb83-75a58cf3c396	897ba9f3-35dd-4150-94c2-523c3919f164	disabled	Close	Close

SUBSISTEMAS DE FALLA

id	fail_system	port_number	device_id	status
9c3d7570-5316-4427-a609-7ff5e8b9387a	ATS	0	b8ce177e-9aa8-4606-bb83-75a58cf3c396	disabled
28b4a6f0-cccd-4d26-970b-36e140512281	HASLER	2	b8ce177e-9aa8-4606-bb83-75a58cf3c396	disabled
386bceb5-05d7-4532-8019-082a2cf85700	BRAKES	1	b8ce177e-9aa8-4606-bb83-75a58cf3c396	disabled

Fig. 3: Modelo de datos de los dispositivos y sus subsistemas de fallas.

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* que oficia de orquestador entre el cliente, el dispositivo *SAL/T* que publica los mensajes y *Apache Kafka* [9], un sistema distribuido que realiza la lectura de la información y su posterior procesamiento.

Dado que los mensajes intercambiados entre las partes contienen información sensible, se ha optado por agregar la capa de seguridad *TLS* [10]. En este sentido, resulta indispensable utilizar el certificado *X509* [11] para prevenir ataques del tipo *adversary-in-the-middle* [12] y utilizar certificados emitidos por autoridades reconocidas.

C. Capa de almacenamiento

La capa de almacenamiento 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 relacional *PostgreSQL* [13] 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.

En la figura 2 se observa el modelo de datos que representa a los dispositivos SAL/T, las formaciones y las entidades. El primero cuenta con un identificador del dispositivo en el sistema, el estado de operación y una referencia al tren en el que se desplegó el artefacto. El modelo de la formación está enriquecido con la entidad a la que pertenece y el numero de serie del tren. Por último, las entidades están compuestas por nombre y descripción. Luego, en la imagen 3 se aprecian los subsistemas de fallas relacionados con el dispositivo SAL/T. Su modelo esta integrado con el nombre del subsistema, el id del dispositivo al que está asociado y el estado actual del mismo.

D. Capa de interacción

La capa de interacción se encuentra conformada por tres unidades. Por un lado, se ha desarrollado en el lenguaje *Kotlin* [14] en conjunto con el *framework Ktor* [15], 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* [16], el cual se encuentra conectado a una base de datos relacional (*PostgreSQL*) basado en el lenguaje de consulta estructurado *SQL* [17], que permite exponer una *API GraphQL* [18] a aquellos clientes que deseen obtener una visualización del modelo de datos propuesto en que se conjuga la información de cada formación ferroviaria 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 visualización

En esta capa, se ha utilizado el lenguaje de programación *TypeScript* [19] junto con la biblioteca gráfica *React* [20] para brindar una web reactiva en la 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 ferroviaria a la que pertenece cada usuario.

En la figura 4 se puede apreciar el dashboard de la central operativa al que tendrán acceso los operadores de cada entidad. En el mismo se observan distintas tarjetas con el estado de cada subsistema de fallas y del tren, como así también un velocímetro que informa la velocidad de la formación.

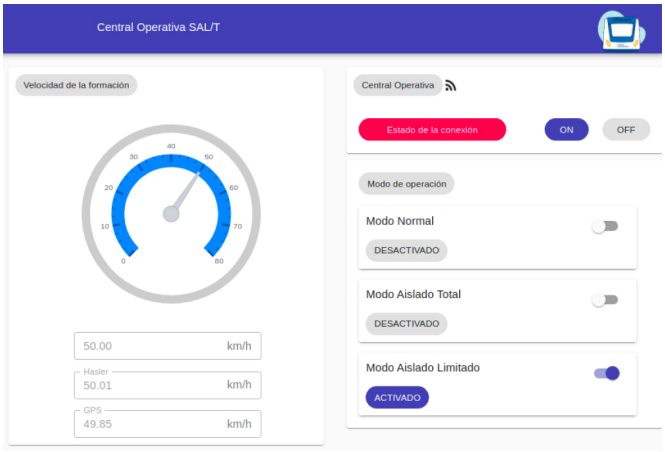

Fig. 4: Panel de control de la central operativa.

IV. Conclusiones

Los avances obtenidos permiten confirmar que el diseño propuesto ha sido el apropiado.

La arquitectura implementada facilitó la disponibilidad y la explotación de la información proveniente de los dispositivos SAL/T, así como del microservicio correspondiente a la autenticación de usuarios. De este modo, se posibilitó la visualización de los integrantes y los dispositivos de cada entidad ofreciendo una solución sencilla y escalable.

En la actualidad, se encuentra en desarrollo propiciar al operario el envio de comandos para el control de los dispositivos activos que este tenga asignados, actualizar la configuración de los parámetros de cada dispositivo y finalmente, brindar la seguridad en las comunicaciones agregando la capa *TLS* en el protocolo *MQTT*.

Referencecias

[1] 'Sistema de supervisión de la seguridad del material ferroviario utilizando patrones de diseño', Ivan Mariano Di Vito, Pablo Gomez, Ariel Lutenberg, Libro de trabajos del CASE2019, Congreso Argetino de Sistemas Embebidos, Santa Fe, Argentina. (2019).

[2] CONICET-GICSAFe (June 2022). [Online]. Disponible: https://sites.google.com/view/conicet-gicsafe/inicio

[3] Trenes Argentinos (June 2022). [Online]. Disponible: https://www.argentina.gob.ar/transporte/trenes

[4] 'Internet of Things for Measuring Human Activities in Ambient Assisted Living and e-Health', Amine Rghioui, Sendra Sandra, Lloret Jaime, Oumnad Abedlmajid, Network Protocols and Algorithms. (2016).

[5] HTTP (June 2022). [Online]. Disponible: https://developer.mozilla.org/es/docs/Web/HTTP

[6] MQTT (June 2022). [Online]. Disponible: https://mqtt.org/

[7] QoS (June 2022). [Online]. Disponible: https://www.hivemq.com/blog/mqtt-essentials-part-6-mqtt-quality-of-service-levels/

[8] NUCLEO-F429ZI (June 2022). [Online]. Disponible: https://www.st.com/en/evaluation-tools/nucleo-f429zi.html

[9] Kafka (June 2022). [Online]. Disponible: https://kafka.apache.org/

[10] 'TLS/SSL - MQTT Security Fundamentals' (2022). [Online]. Disponible: https://www.hivemq.com/blog/mqtt-security-fundamentals-tls-ssl/

[11] X509 (June 2022). [Online]. Disponible: https://www.ssl.com/faqs/what-is-an-x-509-certificate/

[12] Adversary-in-the-Middle (June 2022). [Online]. Disponible: https://attack.mitre.org/techniques/T1557/

[13] PostgreSQL (June 2022). [Online]. Disponible: https://www.postgresql.org/

[14] Kotlin (June 2022). [Online]. Disponible: https://kotlinlang.org/

[15] Ktor (June 2022). [Online]. Disponible: https://ktor.io/

[16] Hasura (June 2022). [Online]. Disponible: https://hasura.io/

[17] SQL (June 2022). [Online]. Disponible: https://www.w3schools.com/sql/

[18] API GraphQL (2022). [Online]. Disponible: https://graphql.org/

[19] TypeScript (June 2022). [Online]. Disponible: https://www.typescriptlang.org/

[20] React (June 2022). [Online]. Disponible: https://reactjs.org/