TecnoCampus
Escola Superior
Politècnica

Centre adscrit a la
upf. Universitat Pompeu Fabra Barcelona

Grau en Enginyeria Informàtica de Gestió i Sistemes d'Informació

Connectivitat remota i resolució d'equips a AlphaMonitor

Memòria

IVAN FRUTOS ZURITA

TUTOR: ALFONS PALACIOS GONZÁLEZ

2023-2024

TecnoCampus
Mataró-Maresme

Abstract

This final degree project focuses on enhancing an existing monitoring platform, AlphaMonitor, to enable remote, autonomous, and automatic management of connections and error resolution in remote equipment. The project aims to bolster the security of these systems while improving operational efficiency. Through the implementation of new functionalities, the already existing tool for diagnosing the status of installed equipments called AlphaMonitor empowers itself to address errors in remote equipment seamlessly, ultimately enhancing the reliability of services provided by the company Alphanet.

To achieve this objective, a comprehensive study of the current system's functionality was conducted, followed by regular meetings with technicians to understand project requirements and specific needs. Subsequently, a detailed analysis was carried out to identify all relevant use cases, paving the way for the coding phase.

This project is of paramount importance for Alphanet as it addresses a critical requirement within its business model. The integration of new functionalities into AlphaMonitor does not only improve operational efficiency but also enhances the security of remote equipment, providing added value to clients.

The benefits include streamlined error resolution, enhanced security measures, and increased client satisfaction.

Resum

Aquest projecte de final de carrera se centra en millorar una plataforma de monitorització existent, AlphaMonitor, per permetre la gestió remota, autònoma i automàtica de connexions i resolució d'errors en equips remots. El projecte té com a objectiu reforçar la seguretat d'aquests sistemes mentre millora l'eficiència operativa. Mitjançant la implementació de noves funcionalitats, l'eina ja existent pel monitoratge de l'estat d'equipaments instal·lats anomenada AlphaMonitor es capacita per abordar els errors en els equips remots de manera coherent, millorant en última instància la fiabilitat dels serveis proporcionats per l'empresa Alphanet.

Per assolir aquest objectiu, es va realitzar un estudi exhaustiu de la funcionalitat del sistema actual, seguit de reunions periòdiques amb els tècnics per comprendre els requisits del projecte i les necessitats específiques. Posteriorment, es va dur a terme una anàlisi detallada per identificar tots els casos d'ús rellevants, obrint pas a la fase de codificació.

Aquest projecte és d'una importància cabdal per a Alphanet ja que aborda un requisit crític dins del seu model de negoci. La integració de noves funcionalitats en AlphaMonitor no només millora l'eficiència operativa sinó que també reforça la seguretat dels equips remots, proporcionant un valor afegit als clients.

Els avantatges inclouen la resolució d'errors simplificada, mesures de seguretat reforçades i una major satisfacció dels clients.

Resumen

Este proyecto de fin de carrera se centra en mejorar una plataforma de monitoreo existente, AlphaMonitor, para permitir la gestión remota, autónoma y automática de conexiones y resolución de errores en equipos remotos. El proyecto tiene como objetivo reforzar la seguridad de estos sistemas mientras mejora la eficiencia operativa. Mediante la implementación de nuevas funcionalidades, la ya existente herramienta de monitorización de equipos instalados AlphaMonitor se capacita para abordar los errores en los equipos remotos de manera coherente, mejorando en última instancia la fiabilidad de los servicios proporcionados por la empresa Alphanet.

Para lograr este objetivo, se realizó un estudio exhaustivo de la funcionalidad del sistema actual, seguido de reuniones periódicas con los técnicos para comprender los requisitos del proyecto y las necesidades específicas. Posteriormente, se llevó a cabo un análisis detallado para identificar todos los casos de uso relevantes, abriendo paso a la fase de codificación.

Este proyecto es de una importancia capital para Alphanet ya que aborda un requisito crítico dentro de su modelo de negocio. La integración de nuevas funcionalidades en AlphaMonitor no solo mejora la eficiencia operativa sino que también refuerza la seguridad de los equipos remotos, proporcionando un valor añadido a los clientes.

Los beneficios incluyen la resolución de errores simplificada, medidas de seguridad reforzadas y una mayor satisfacción de los clientes.

Índex

Índex de figures

Glossari de termes

ALPR	Automatic License Plate Recognition
ADM	AlphaDataManager
AM	AlphaMonitor
AP	ApiPulse
AWS	Amazon Web Services
BD	Base de dades
CPU	Unitat de processament central d'un ordinador
DNS	Sistema de noms de domini
HEARTBEAT	Senyal periòdica enviada per un dispositiu que indica l'estat del seu funcionament
JPA	Java Persistence API
JSON	JavaScript Object Notation
KPI	Indicador clau de rendiment
MVC	Model vista controlador
PING	Protocol de xarxa que s'utilitza per a comprovar la connexió entre dos dispositius
RDS	Relational Database Service, d'AWS
SAT	Servei d'Assistència Tècnica
S3	Simple Storage Service, d'AWS
SQS	Simple Queue Service, d'AWS

1. Introducció

Aquest és un projecte proposat per l'empresa Alphanet, líder en solucions tecnològiques per a la seguretat ciutadana. L'objectiu d'aquest treball és millorar la plataforma AlphaMonitor mitjançant el desenvolupament i implementació de noves funcionalitats destinades a gestionar de manera remota, autònoma i automàtica les connexions i la resolució d'errors en equips de monitoratge distribuïts en àmbits urbans i viaris. Aquesta millora té com a finalitat principal augmentar la seguretat dels equips, optimitzar els processos operatius, elevar el nivell de disponibilitat dels equips i incrementar la fiabilitat dels serveis proporcionats per Alphanet.

Per dur a terme aquest projecte, s'han seguit una sèrie de passos:

- Anàlisi exhaustiu del sistema de monitoratge existent per comprendre el seu funcionament i les necessitats operatives.
- Definició clara del projecte, establint els objectius, l'abast i la llista de tasques necessàries, així com els requeriments tecnològics i funcionals.
- Avaluació de la viabilitat del projecte per assegurar-ne la factibilitat tècnica i econòmica.
- Identificació i definició dels casos d'ús, juntament amb l'elaboració de diagrames de casos d'ús per visualitzar els processos.
- Disseny del model de la base de dades per a un emmagatzematge eficient de la informació rellevant.
- Implementació i codificació de les noves funcionalitats previstes, seguint les especificacions i els requeriments establerts.

Les principals característiques d'aquest projecte inclouen:

- La detecció i gestió dels determinats dispositius que presentin fallades.
- La capacitat de notificar i resoldre automàticament aquests incidents mitjançant una resposta eficient, ràpida i eficaç per part del mateix sistema.
- La generació de gràfics estadístics per a una visualització de les accions realitzades sobre els equips.
- La integració completa amb la plataforma AlphaMonitor.

Aquest enfocament permet millorar la resposta i resolució de les falles en els equips remots, contribuint a la fiabilitat i eficiència dels serveis de monitoratge d'Alphanet i reforçant la seva posició com a empresa referent en tecnologies per a la seguretat ciutadana.

2. Marc teòric

2.1 Alphanet

Alphanet [1] és una empresa catalana amb setze anys de recorregut que s'especialitza en solucions de trànsit per administracions i cossos policials. Les solucions d'Alphanet es troben implementades en diversos països i a més de dos-cents municipis. El sector principal en què desenvolupa la seva activitat Alphanet és el sector de la seguretat, encara que es troben diferents àmbits d'actuació: seguretat ciutadana, seguretat vial, mobilitat i medi ambient.

- Seguretat Ciutadana: es col·labora estretament amb els cossos de policia a actuar diligentment, proporcionant una millor resposta davant dels perills, prevenint delictes i facilitant la resolució d'investigacions criminals.
- Mobilitat: es proporcionen solucions pel control d'emissions contaminants en zones urbanes, ajudant a establir una millor relació entre vehicles i vianants, pacificant els vials i zones estratègiques.
- Smart Analytics: es realitzen anàlisi intensius de dades que possibiliten conèixer i comprendre les actuacions dels vehicles, les emissions de diòxid de carboni CO2 i el parc automobilístic. També es duen a terme estudis de tendències que faciliten la comprensió de les dades que reflecteixen la realitat dels municipis, tot contribuint a la correcta presa de decisions informades.
- Medi Ambient: s'implementen solucions que ajuden els municipis a assolir els grans reptes mediambientals, com la reducció de la contaminació atmosfèrica i millora de la qualitat de l'aire, transició a un model de mobilitat sostenible, millorar la gestió dels residus i reducció dels abocaments il·legals.

El sector de la seguretat ciutadana és una àrea crítica en qualsevol societat, ja que busca garantir la seguretat i el benestar dels ciutadans. Aquest sector inclou una àmplia gamma d'activitats i serveis, com ara la policia, els serveis d'emergència, la prevenció del crim, la gestió de desastres, la seguretat cibernètica i la videovigilància.

En molts països, el sector de la seguretat ciutadana està experimentant un augment de la demanda i de la inversió, especialment en àrees urbanes densament poblades. Això es deu a

diversos factors, com ara l'augment de la criminalitat, els reptes relacionats amb el terrorisme, la necessitat de millorar la resposta a les emergències i la creixent preocupació per la seguretat cibernètica.

En el context de la videovigilància de seguretat ciutadana, hi ha diverses empreses que ofereixen solucions i serveis en aquest àmbit.

Algunes de les empreses més conegudes inclouen:

1. Axis Communications: És una empresa líder en la fabricació de càmeres de vigilància de xarxa, que ofereixen solucions avançades per a la seguretat ciutadana, inclosos sistemes de videovigilància d'última generació.
2. Hikvision: És una de les empreses líders en el mercat mundial de sistemes de videovigilància, tenen una àmplia gamma de productes com càmeres de seguretat, sistemes de vídeo vigilància IP i analítiques de vídeo.
3. Bosch Security Systems: És una altra empresa reconeguda en el sector de la seguretat ciutadana, ofereixen una àmplia gamma de productes i solucions, inclosos sistemes de videovigilància, control d'accés i detecció d'incendis.
4. Dahua Technology: És una empresa destacada en la fabricació de càmeres de vigilància de seguretat i sistemes de videovigilància, que ofereixen solucions innovadores per a la seguretat ciutadana.

Amb la creixent demanda de seguretat ciutadana, s'espera que aquest sector continuï expandint-se i evolucionant per abordar les necessitats emergents de les comunitats i les ciutats.

2.2 Plataformes d'Alphanet

Alphanet té tres plataformes per desenvolupar la seva activitat; l'AlphaDataManager [2], l'AlphaMonitor i l'AlphaAlert [3].

2.2.1 AlphaDataManager

AlphaDataManager és una plataforma allotjada al núvol basada en la gestió i explotació de dades, que transforma les dades captades pels equips de detecció de matrícules en informació útil.

Té responsabilitats relacionades amb:

- Seguretat ciutadana: investigacions policials i anàlisi de dades.
- Seguretat viària: controls de permanència, controls d'accés, controls semafòrics i gestió de sancions.
- Smart Analytics: informar del trànsit, emissions, identificació de tendències.

2.2.2 AlphaMonitor

AlphaMonitor és una plataforma allotjada al núvol per ús intern de l'empresa, s'usa per monitorar els sistemes de reconeixement de matrícules i el seu estat. La seva missió és ajudar l'equip de SAT o tècnics de l'empresa a muntar dispositius i assistir als tècnics de camp, tenir registres fiables de les instal·lacions amb les dades correctes de les mateixes incloent una millora del temps de resposta de l'empresa gràcies a la detecció d'avaries. En definitiva, millora la eficiència de l'empresa alhora que millora la disponibilitat del sistema de detecció de matrícules.

Té responsabilitats relacionades amb:

- Prevenir avaries: ajuda a anticipar futures possibles avaries.
- Manteniment: ajuda com a eina de gestió del manteniment dels equips instal·lats i com a eina de control d'inventari pels tècnics.
- Monitoratge: permet veure el funcionament en temps real dels equips instal·lats i entendre les dades que arriben.
- Sistema d'alertes: permet generar avisos d'aquells dispositius instal·lats que no estan tenint el comportament esperat, i que per tant no estan funcionant correctament.

2.2.3 AlphaAlert

AlphaAlert és una aplicació mòbil que permet als clients d'Alphanet fer ús de la plataforma AlphaDataManager en dispositius mòbils. Està dissenyada perquè els usuaris puguin utilitzar la plataforma mentre es troben de servei als carrers dels municipis, rebre alertes i disposar d'accés immediat a la informació de valor que necessitin.

2.3 Arquitectura de software

Alphanet té la seva arquitectura de software estructurada de forma que les plataformes principals com AlphaDataManager i AlphaMonitor es recolzen en una pila de serveis que els complementen i els fan funcionar de manera més eficaç i eficient.

La complexitat de l'arquitectura de software d'Alphanet pot semblar una font de confusió o una despesa de recursos innecessària. No obstant això, aquesta estructura meticulosament dissenyada és en realitat essencial per a garantir el funcionament robust i la durabilitat a llarg termini de les plataformes principals, com AlphaDataManager i AlphaMonitor a part de permetre a cada servei enfocar-se en la seva responsabilitat, arribant a abaratir costos.

La integració de múltiples serveis en una pila ben definida no només ajuda a mantenir les responsabilitats clarament diferenciades, sinó que també permet una millor escalabilitat, flexibilitat i mantenibilitat del sistema en conjunt. Cadascun d'aquests serveis, tot i ser aparentment petits i específics, contribueix de manera crítica al funcionament harmoniós de l'ecosistema d'Alphanet.

Encara que aquesta aparent complexitat inicial pugui semblar desafiadora, és precisament aquesta estructura que assegura la solidesa i l'evolució constant de la plataforma al llarg del temps.

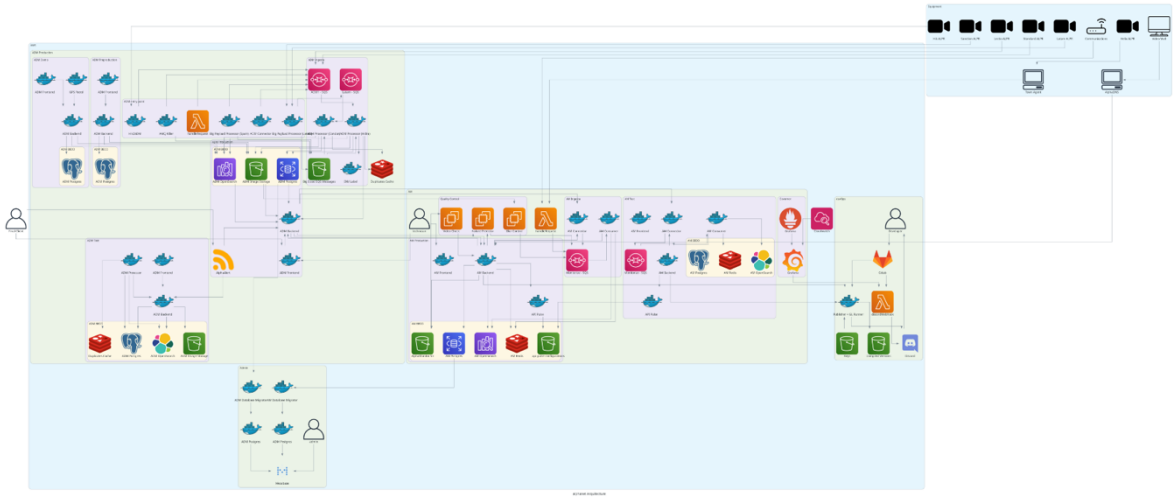

Figura 1. Infraestructura dels serveis d'Alphanet. Font: Alphanet.

2.3.1 AlphaMonitor

L'AlphaMonitor és l'encarregat de monitoritzar els equips remots instal·lats als diferents municipis clients d'Alphanet, el backend és responsable d'actualitzar l'estat de l'equipament. Per decidir l'estat de l'equipament, utilitza un sistema d'escalat.

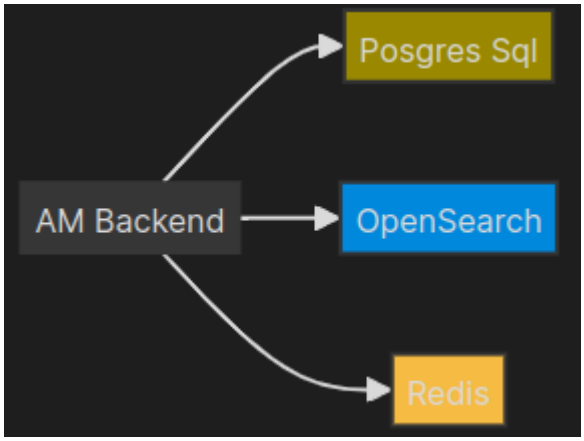

Figura 2. Serveis interconnectats que possibiliten el sistema de monitoratge. Font: Alphanet.

El sistema comença monitorant les MonitorProperties, que inclouen HEARTBEAT, PING, ALPR i ALPR_HELIX.

Agafa els registres de la base de dades postgres de la taula MonitorProperty i recupera el camp valor de cada registre. El camp valor defineix durant quants minuts es pot considerar vàlida la darrera missatge rebut per part del consumidor que contingués dades correctes.

Per a cada registre de MonitorProperty, busca a Redis utilitzant el seu ID, SERIAL o ID MONITOR (l'usuari del monitor és responsable de seleccionar com es monitoritza i quin tipus d'ID s'utilitza) i recupera la data del darrer registre. Si està dins del límit de temps, el camp Fase de la propietat del monitor es defineix com a (1), que significa OK. Si no hi ha dades recents, es defineix com a IRREGULAR(2), i si ja era irregular, es defineix com a ALERT (3).

La implementació del codi permet aplicar lògiques diferents per a cada tipus de propietat del monitor. Per exemple, en el cas de l'ALPR, quan és de nit, el temps vàlid pot ser estès.

2.3.1.1 AM Backend

El backend d'AlphaMonitor, desenvolupat amb Spring Boot i Java 17, es presenta com el component principal que impulsa la lògica i l'orquestració de dades de l'aplicació. Garanteix el funcionament i el monitoratge sense costures dels equips de xarxa mitjançant la integració de diverses funcionalitats i serveis.

A continuació, es presenta un resum concís de les seves característiques principals i l'arquitectura:

- Funcionalitat de l'API:
    - Gestió de l'inventari d'equips: Mitjançant una API ben definida, el backend facilita la creació, actualització i seguiment de l'inventari d'equips, permetent una integració sense costures i una supervisió dels dispositius de xarxa.
- Emmagatzematge de Dades:
    - Base de dades RDS PostgreSQL: Emmagatzema informació detallada de l'inventari, incloent-hi les especificacions dels dispositius, els registres de manteniment i l'estat operatiu, assegurant que les dades estiguin organitzades sistemàticament i siguin fàcilment accessibles.

- Emmagatzematge d'Amazon S3: Utilitza S3 per a l'emmagatzematge segur i escalable d'imatges, documents i fitxers adjunts relacionats amb els equips i les seves ubicacions físiques. Això garanteix una alta disponibilitat i redundància dels fitxers crítics.

- Recuperació i Anàlisi de Dades:
  - Base de dades OpenSearch: Empra OpenSearch per a la recuperació eficient de dades analítiques, com gràfics de rendiment i tendències històriques, permetent preses de decisions informades i una gestió proactiva.
  - Redis per a Monitoratge en Temps Real: Aprofita Redis per al monitoratge en temps real de l'estat dels equips. Actualitza de manera contínua els estats dels dispositius basant-se en *feeds* de dades en viu, assegurant respostes oportunes als canvis o problemes operatius.
- Integració i comunicació:
  - Comunicació amb Serveis Interns i Externs: El backend facilita la comunicació tant amb serveis interns (dins de la infraestructura d'AlphaMonitor) com amb externs, com TownAgent o AlphaDNS. Aquesta interoperabilitat és crucial per a un ecosistema cohesionat i millora la capacitat de la plataforma per a respondre a condicions de xarxa dinàmiques.

Per concloure, el backend d'AlphaMonitor integra diverses tecnologies i serveis per a proporcionar una gestió i monitoratge comprensius dels equips de xarxa. Aprofitant l'emmagatzematge en núvol, les solucions de bases de dades i el processament de dades en temps real, garanteix la fiabilitat i l'eficiència de les operacions de xarxa.

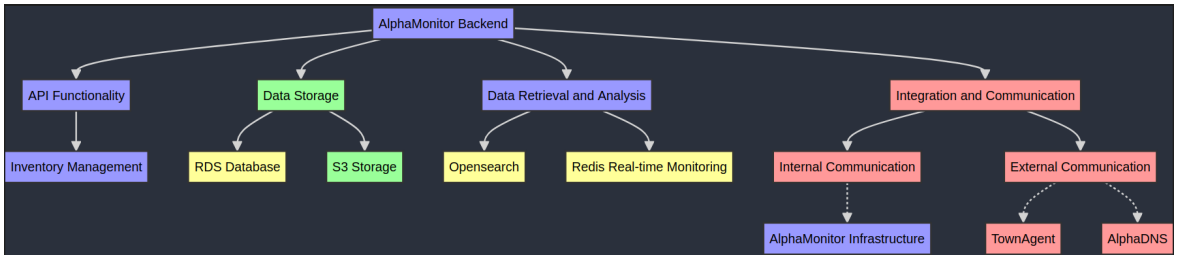

Figura 3. Backend d'AlphaMonitor i les seves responsabilitats juntament amb els serveis que les possibiliten. Font: Alphanet.

2.3.1.2 AM Frontend

El frontend d'AlphaMonitor és possible gràcies a l'ús de tecnologies com Angular 12, TypeScript i és estilat amb Bootstrap i la llibreria Stoke.

Alguns punts destacables del Frontend d'AlphaMonitor són:

- Autenticació:
  - Gestió de sessions segura: El frontend utilitza autenticació basada en *tokens* per gestionar les sessions d'usuari. Aquest mètode garanteix que les interaccions del client amb el backend estiguin autenticades i siguin segures.
- Control d'accés basat en permisos:
  - Permisos de Grup d'Usuaris: Els usuaris s'assignen a diferents grups, cadascun amb permisos específics que determinen els nivells d'accés a diverses seccions de la plataforma, com ara el tauler de control, el mapa, la gestió de tiquets i altres.
  - Visualització Dinàmica del Menú: El menú de navegació s'ajusta dinàmicament per reflectir els permisos de l'usuari connectat, garantint que els usuaris només vegin opcions rellevants segons els seus drets d'accés.
- Visualització dels gràfics:
  - ECharts: ECharts s'utilitza per crear gràfics en els fronts AlphaMonitor i AlphaDataManager. Aquests gràfics es troben a la carpeta compartida d'Angular.

Aquest projecte ha d'aprofitar el control d'accés basat en permisos i implementar nous permisos, atès que no tots els usuaris han de poder realitzar accions sobre equips o editar marques i models, entre d'altres accions.

2.3.2 Serveis complementaris d'AlphaMonitor

AlphaMonitor es recolza en altres serveis i tecnologies que el complementen i el milloren, de forma que cada servei té la seva responsabilitat dintre del funcionament global del sistema.

2.3.2.1 AM Connector

El Connector d'AM, desenvolupat utilitzant Spring Boot i Java 21, actua com a servei intermediari que s'interfereix amb diverses fonts de dades per a recuperar informació, la qual després envia al servei Amazon Simple Queue Service (SQS) per a ser consumida pel Consumidor d'AM. És el pont d'informació entre ADM i AM.

Responsabilitats clau:

- Recuperació de dades: És responsable d'obtenir dades d'altres serveis, com ara AlphaDataManager. Per exemple, obté dades de captura de matrícules del sistema de Reconeixement Automàtic de Matrícules (ALPR) amb propòsits de monitoratge d'equips, representació de gràfiques...

2.3.2.2 AM Consumer

El Consumidor d'AM, desenvolupat amb Spring Boot i Java 21, és una part integral de la infraestructura d'AlphaMonitor, encarregat de processar els missatges entrants de càmeres, *routers* i aplicacions que informen sobre esdeveniments relacionats amb els equips.

Aquest component sofisticat està dissenyat per garantir un maneig eficient dels missatges mitjançant la implementació del patró de disseny estratègic per al processament dinàmic.

A continuació, s'aprofundeix en les seves funcionalitats i el flux operatiu, incloent la interacció amb el Backend d'AM:

- Processament avançat de missatges:
    - Utilització del patró estratègic: El Consumidor d'AM fa servir el patró estratègic per a navegar a través d'una seqüència de manegadors per trobar un capacitat de processar el missatge entrant. Aquest enfocament garanteix flexibilitat i adaptabilitat en el processament de missatges.
    - Aplicació de l'estratègia en paral·lel: Aplica el patró estratègic en un entorn multitasca, permetent el processament simultani de múltiples missatges, cada un potencialment requerint una estratègia de tractament diferent.

- Emmagatzematge selectiu de dades: Les dades processades s'emmagatzemen estratègicament a Redis, OpenSearch o a ambdós, adaptats al contingut i la importància del missatge, optimitzant la recuperació i l'anàlisi de dades.
- Interacció amb el Backend: Per a certs missatges que requereixen anàlisis especialitzades, el Consumidor d'AM fa crides a punts d'extrem específics del Backend d'AM. Aquesta interacció és mínima i està dissenyada per a ser eficient, garantint que no afecti significativament el temps de processament global.

Dades destacades del funcionament:

- Sondeig d'Amazon SQS: El Consumidor revisa periòdicament Amazon SQS per a nous missatges, aprofitant la gestió del servei en la lliurament i processament dels missatges.
- Aplicació del Patró Estratègic: Empra el patró estratègic per identificar el manegador adequat per a cada missatge, garantint un processament flexible i precís.
- Manipulació Eficient de Dades: Després del processament, les dades s'emmagatzemen a Redis, OpenSearch o a ambdós, basant-se en criteris predeterminats. Per a necessitats específiques de traducció de dades, interactua amb el Backend d'AM, tot i que això es manté al mínim per mantenir l'eficiència de processament.

Aquesta visió general subratlla el paper fonamental del Consumidor d'AM en el processament i la gestió dels missatges relacionats amb l'equip dins de la infraestructura d'AlphaMonitor. Mitjançant la integració intel·ligent de patrons de disseny, serveis en núvol i interaccions selectives amb el backend, es consolida com a pedra angular per a la monitorització i gestió efectives d'equips de xarxa, posant l'èmfasi en l'eficiència i la fiabilitat en les operacions.

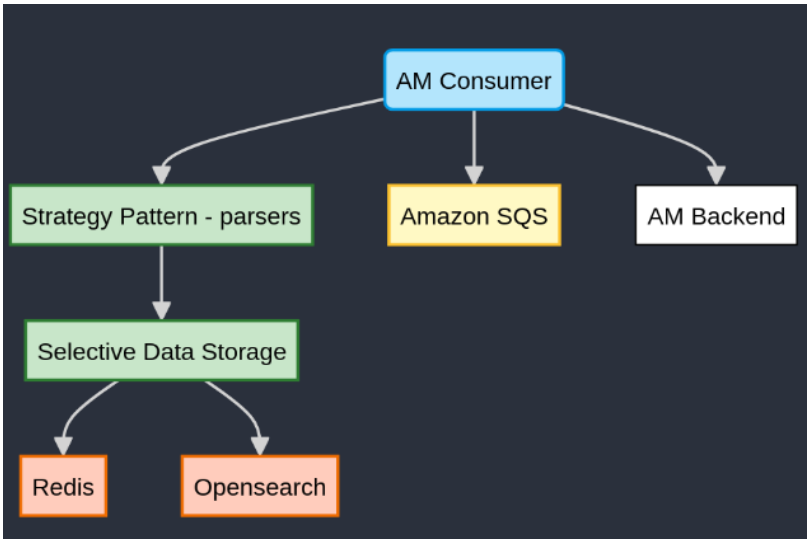

Figura 4. Esquema de l'AM Consumer i les seves relacions. Font: Alphanet.

2.3.2.3 ApiPulse

L'ApiPulse és el servei creat per necessitat d'aquest projecte. Impulsat amb Java 21, és l'encarregat d'executar les peticions de les accions rebudes de l'AlphaMonitor i de processar-les segons els requisits.

Aquest servei té la responsabilitat de determinar el tractament adequat per a cada acció rebuda. Depenent dels requeriments, pot prendre diverses accions, com ara enviar dades a S3 d'Amazon, retornar informació a l'AlphaMonitor o enviar-les al Consumidor d'AM perquè siguin processades i acabin a l'OpenSearch.

Mitjançant aquesta estructura, l'ApiPulse es converteix en un element clau en la gestió i processament de les dades en el context de l'ecosistema d'AlphaMonitor, assegurant un flux eficient i adequat de la informació per a suportar les operacions i la presa de decisions.

2.4 Antecedents

Els antecedents indiquen com els tècnics han de fer un esforç de raonament lògic i memòria a l'hora de connectar-se als equips per solucionar els problemes, indicador de que és precís disposar d'aquesta informació que facilita de manera entenedora i fàcil la plataforma AlphaMonitor.

És estrictament necessari guardar nova informació dels equips a la base de dades per tal que aquesta pugui ser recuperada per l'AlphaMonitor, concretament el port amb una columna nova a la taula d'equipaments, l'històric d'intents de reinici i els èxits de reinici.

Pel que a la part de la API respecta, també és precís que l'AlphaMonitor pugui oferir certes accions segons el fabricant d'aquell equip, i que sàpiga quines accions ha de permetre i com les ha de dur a terme segons la API de l'equip de manera interna, de forma que el tècnic que ha d'operar amb l'equip pot veure les accions disponibles i executar-les, encarregant-se de gestionar la API de manera interna únicament amb la plataforma i les opcions que aquesta ofereix. Aquest fet indica que també és necessari tenir aquesta informació a la base de dades.

Tota la feina esmentada conjuntament amb el codi necessari tant al frontend com al backend de l'AlphaMonitor han de fer possible assolir els objectius d'aquest projecte de manera eficient i satisfactòria.

2.5 Motius del canvi

Fins al moment de la realització d'aquest projecte, se segueix un procés manual per part dels tècnics de l'equip SAT per corregir errors detectats pel sistema de monitoratge AlphaMonitor. Aquest procediment implica l'accés remot als equips, on els tècnics han de recordar i aplicar manualment els paràmetres de connexió, incloent-hi el port i l'API, amb la complexitat afegida de canvis segons el fabricant de l'equip.

Amb l'ús d'aquesta metodologia, els tècnics inverteixen un temps excessiu en establir connexions i operar fins a la resolució dels errors, generant una dedicació de temps considerable per a cada equip amb problemes.

2.6 Tendències

A l'actualitat existeix una creixent tendència a l'ús de tecnologies d'aprenentatge autònom i automàtic així com d'anàlisi de dades per a millorar aspectes clau com efectivitat, eficiència i nivell de precisió dels sistemes de monitoratge de càmeres de trànsit i altres equipaments tecnològics relacionats amb la captació d'informació, la transmissió de dades i les telecomunicacions.

En el context d'una empresa que pertany al sector de la seguretat ciutadana i està fortament lligada amb la tecnologia, és precís millorar els processos interns que puguin fer-se de manera autònoma i automàtica. Per això mateix i tenint en compte els antecedents disponibles, es poden definir els objectius d'aquest projecte i la necessitat d'informació que cal per dur-lo a terme, concretant també si és possible la seva realització mitjançant un estudi de viabilitat econòmica, tècnica i mediambiental.

2.7 Triatge tecnològic

El procés de selecció tecnològica, especialment pel que fa al software i les versions, s'ajusta a les tecnologies ja implementades a l'empresa. Això s'orienta cap a l'eficiència i la maximització de la compatibilitat entre tots els serveis de la infraestructura empresarial. Pel que fa a les connexions amb les càmeres per dur a terme accions sobre elles, és imperatiu utilitzar crides HTTP ja que és el protocol establert pel fabricant. L'única altra opció és utilitzar softwares específics de cada fabricant, cosa que no s'alinea amb l'objectiu d'implementar una solució universal vàlida per a tots els fabricants existents.

3. Objectius i abast

Aquest projecte és un projecte desenvolupat per obtenir millores internes a l'empresa, que a la vegada produeix un impacte positiu envers els clients reals. Per tant, pot fer-se una distinció entre el client directe i el client indirecte d'aquest projecte.

3.1 Clients

3.1.1 Client directe

El client directe d'aquest treball és tot l'equip intern de tècnics SAT, que fan ús diari de la plataforma sobre la que es volen integrar aquestes funcionalitats, que és la plataforma interna ja existent AlphaMonitor.

3.1.2 Client indirecte

Els clients indirectes són els organismes governamentals, forces policials i ajuntaments que mantenen contractes amb l'empresa. Aquests clients experimenten un notori augment en el seu nivell de satisfacció gràcies a la plataforma AlphaMonitor.

3.2 Objectius del client

- Millorar el temps de resposta sota la resolució d'incidències en equips.
- Millorar el grau de satisfacció dels clients.
- Millorar la seguretat i la protecció de dades.
- Fer ús únicament de l'AlphaMonitor per gestionar connexions remotes sobre els equips.
- Integrar les noves funcionalitats amb la plataforma i la xarxa d'equips existents.

3.3 Objectius del projecte

- Permetre realitzar reinicis, actualitzacions de configuració i peticions d'obtencions de dades als equips instal·lats de manera remota i automàtica.
- Analitzar les dades recollides per la pròpia plataforma AlphaMonitor, manegar-les, interpretar-les i prendre accions de manera autònoma en base a aquestes.
- Implementar les noves funcionalitats per tal que els clients puguin usar-les mitjançant únicament l'ús de l'AlphaMonitor.
- Oferir una interfície d'usuari accessible, usable i entenedora per als clients.
- Augmentar la seguretat en l'accés als equips instal·lats.

La implementació de les noves funcionalitats d'aquest projecte a la plataforma AlphaMonitor no només optimitza l'eficiència operativa en resoldre problemes de manera autònoma, sinó que també permet als tècnics centrar-se en proporcionar un servei de major qualitat als clients, contribuint inevitablement a la millora global de la seva satisfacció.

Aquest fet passa per una millora substancial de l'AlphaMonitor, convertint-lo en un sistema de presa de decisions, que adquireix aquesta habilitat i pren decisions basades en els resultats en temps real de monitoratge capaç de resoldre problemes de manera autònoma i automàtica, proporcionant una major agilitat en la resolució de problemes amb els equips

Aquesta millora es tradueix en un benefici doble: primerament, es tradueix a una reducció del temps a realitzar operacions en els equips per part dels tècnics, ja que la plataforma AlphaMonitor s'encarrega eficientment de moltes tasques. En segon lloc, s'elimina la necessitat d'establir manualment connexions, fet pel qual els tècnics poden centrar-se en atendre les avaries realment complexes i els clients de manera més efectiva.

3.4 KPIs

Els indicadors clau de rendiment són mètriques que mesuren el rendiment i l'èxit d'un projecte, procés o activitat. Ajuden a avaluar el progrés, identificar àrees de millora i prendre decisions informades per assolir els objectius. Proporcionen una manera concreta de comprendre l'impacte de les accions i ajustar la estratègia segons les dades.

Els KPIs adequats per avaluar l'èxit d'aquest projecte podrien incloure:

1. Increment d'equips en fase correcta de monitoratge: Mesura l'augment del nombre d'equips en fase correcta (OK) de l'estat de monitoratge. Un nombre major d'equips en aquesta fase indica que s'està aconseguint un nivell més alt de disponibilitat i funcionament dels equips remots instal·lats.
2. Temps de Resolució d'Incidències: Mesura el temps mitjà necessari per solucionar els problemes detectats pels equips remots. Un temps de resolució més curt indica una millora en l'eficiència del sistema.
3. Nombre d'Accions Automàtiques Realitzades: Compta la quantitat d'accions automàtiques que realitza el sistema per corregir errors en equips remots sense intervenció humana. Un augment en aquest nombre reflectiria una millora en l'autonomia i l'eficàcia de la plataforma.
4. Grau de Reducció de Visites Tècniques: Avalua la disminució en el nombre de visites tècniques presencials necessàries per resoldre problemes als equips remots després de la implementació del nou sistema. Menys visites indiquen una millora en la gestió remota de les incidències.
5. Grau de Satisfacció del Client: Mesura la percepció dels clients sobre la qualitat i l'eficàcia del servei després de la implementació del nou sistema. S'ha de realitzar a través de feedbacks, enquestes o altres mètodes de recollida d'opinions.
6. Increment de la Utilització dels Recursos: Avalua si l'ús del sistema ha permès una millor utilització dels recursos de l'empresa, com la reducció de temps del personal tècnic, disminució dels costos de desplaçament o millora de la gestió del temps.
7. Reducció de Costos Operatius: Mesura la reducció de costos associada amb la implementació del nou sistema, incloent-hi els costos de manteniment, les despeses de suport tècnic i els costos derivats de visites presencials.

4. Metodologia

Primerament es duu a terme un procés d'investigació i recerca per a obtenir les dades necessàries, pel que es fa una cerca d'informació en la documentació a nivell intern en l'empresa Alphanet.

Amb aquesta informació s'inicia un procés d'anàlisi i síntesi de la mateixa per tal d'avaluar i comprendre-la, i es poden distingir entre tres tipus d'anàlisi:

- Anàlisi de contingut: aplicar tècniques d'anàlisi de contingut per identificar els conceptes i aspectes principals i les tendències en la informació obtinguda.
- Anàlisi estadístic: fer ús d'eines estadístiques per analitzar les dades i extreure'n informació valuosa.
- Anàlisi de casos: fer un estudi sobre els diferents casos o possibles problemes per tal de dissenyar les millors solucions en tot moment.
- Mapes d'infraestructures: fer ús de mapes conceptuals amb l'objectiu d'aconseguir una millor perspectiva de les relacions entre els diferents serveis relacionats amb el projecte.

Un cop finalitzat el procés d'investigació, anàlisi i síntesi, s'avança i es procedeix a una fase de concepció, disseny i abans d'anar a la execució. Durant aquesta nova etapa les dades recopilades per l'anàlisi previ són usades per tal de confeccionar un pla d'acció específic i adient pel projecte en qüestió amb la missió d'aconseguir els objectius del mateix.

Per tal d'aconseguir-ho, es desglossa el projecte en diferents tasques a assolir, assignant els temps i les prioritats adients a cadascuna d'elles endreçant-les cronològicament en la duració del projecte. Aquesta fase inclou la definició pertinent dels objectius tant a curt com a llarg termini.

Tan bon punt es finalitza la part de concepció, disseny i planificació s'empren eines de gestió de projectes, en aquest cas JIRA Software [4] en col·laboració amb Alphanet, per tal de fer la divisió i assignació correctes de les tasques i aconseguir una traçabilitat òptima durant la evolució del projecte.

Seguidament s'inicia una etapa de desenvolupament i implementació en la que es duu a terme la codificació i integració de les funcionalitats del projecte. Aquest procés consta de

la codificació del backend amb el llenguatge de programació Java a l'entorn de desenvolupament IntelliJ IDEA [5] s'usa la tecnologia Spring Boot [6], la codificació del frontend a IntelliJ IDEA amb Angular [7] i Spring Boot i la de la base de dades amb PostgreSQL [8] a l'eina gestora de base de dades DBeaver [9].

Es freqüenta l'ús de la tecnologia de contenidors Docker [10] en la realització del desenvolupament del codi de backend, permetent les connexions amb la base de dades.

Es treballa amb entregues contínues donat que la fase s'executa amb l'ajuda de l'eina de control de codi GitLab [11].

Després d'acabar aquesta etapa d'execució es duu a terme una fase d'avaluació per tal d'assegurar l'eficàcia i eficiència de les noves funcionalitats del projecte.

4.1 Tasques i *sprints*

Per implementar el projecte seguint l'organització de l'empresa, es duu a terme un treball de planificació i gestió d'organitzar les tasques en *sprints* de dues setmanes, s'han agrupat les tasques en conjunts coherents que poden completar-se en aquest període. A continuació, s'assignen aquests conjunts de tasques a cada *sprint*:

- *Sprint* 1 (20 de desembre - 3 de gener): Durant aquest *sprint*, es focalitza en investigar les APIs dels fabricants de dispositius i analitzar el tractament actual de les dades dels dispositius instal·lats.
- *Sprint* 2 (4 de gener - 17 de gener): En aquest sprint, es centra en investigar els requisits d'integració de les dades en AlphaMonitor i en analitzar els requisits d'emmagatzematge de dades.
- *Sprint* 3 (18 de gener - 31 de gener): Aquí es dedica a investigar els requisits de migració de les dades existents i seleccionar les eines de desenvolupament adequades.
- *Sprint* 4 (1 de febrer - 14 de febrer): Durant aquest sprint, es dissenya l'estructura i el codi per al processament de dades.
- *Sprint* 5 (15 de febrer - 28 de febrer): Es dedica al disseny i desenvolupament de la base de dades.

- *Sprint* 6 (1 de març - 14 de març): Aquí es dissenya i desenvolupa el sistema de resolució remota d'errors en equips.
- *Sprint* 7 (15 de març - 28 de març): En aquest sprint, es centra en el disseny i desenvolupament de la interfície gràfica.
- *Sprint* 8 (29 de març - 11 d'abril): Es realitzen proves amb dades de prova a l'entorn de proves.
- *Sprint* 9 (12 d'abril - 25 d'abril): Finalment, es dedica a la migració de les dades i al disseny i desenvolupament de les interfícies necessàries per realitzar consultes tècniques avançades.

Aquest enfocament permet tenir una visió clara del progrés del projecte i assegura una distribució eficient de les tasques al llarg del temps, mantenint un enfocament iteratiu i adaptable.

5. Definició de requeriments funcionals i tecnològics

Els requeriments funcionals delineen les funcionalitats necessàries per millorar la gestió dels dispositius remots i la resolució d'incidències a través de la plataforma AlphaMonitor. Això inclou la integració de noves dades, el reconeixement dels dispositius, l'avaluació autònoma de les accions i l'enviament de dades a OpenSearch per a l'anàlisi.

Els requeriments tecnològics especifiquen les tecnologies necessàries per aconseguir aquestes funcionalitats, com ara un servei dedicat per a l'execució d'accions, la comunicació eficient entre AlphaMonitor i aquest servei, i l'allotjament del servei en AWS. Aquesta definició és fonamental per garantir la implementació adequada del projecte i el compliment dels objectius establerts.

5.1 Requeriments funcionals

1. Integrar les noves dades que permeten dur a terme les noves funcionalitats en tots els tipus de dispositius ja existents i nous a la plataforma.
2. Reconèixer el dispositiu remot i gestionar el tractament de les seves dades (API, port).
3. Recollir i tractar l'anàlisi d'avaluació de la plataforma sobre l'estat del dispositiu.
4. Avaluar de forma autònoma i automàtica si l'AlphaMonitor ha de realitzar cap acció (reinici) sobre el dispositiu.
5. Enviar les dades necessàries de les execucions a la SQS i a OpenSearch [12].
6. Millorar la seguretat de l'accés als dispositius remots.
7. Consultar l'històric d'accions realitzades sobre equips.
8. Consultar el nombre i tipus de les accions realitzades més rellevants en una nova pantalla dedicada.

5.2 Requeriments tecnològics

1. Aïllar l'execució de les accions sobre els equips a un servei dedicat per tal de mantenir responsabilitats i no saturar la plataforma principal.

2. Aconseguir la correcta comunicació entre AlphaMonitor, que avalua i delega les accions a realitzar, i el nou servei dedicat anomenat ApiPulse, encarregat de rebre i executar aquestes ordres.
3. Aconseguir la correcta comunicació entre un nou servei d'S3 per guardar els fitxers de configuració dels equips i l'ApiPulse.
4. Aconseguir el correcte enviament a la SQS per part de l'ApiPulse.
5. Aconseguir la correcta transformació de dades a la SQS per enviar com es requereix a OpenSearch.
6. Reestructurar la base de dades amb les dades actuals per tal de no perdre dades amb la migració als nous tipus de dades i al canvi de tipus de dades ja existents.
7. Allotjar el nou servei dedicat a l'execució d'accions a AWS [13].
8. Mostrar i visualitzar correctament gràfics de les execucions de les accions sobre els dispositius per obtenir informació i permetre confeccionar millors decisions sobre la seva gestió en base a aquesta.
9. Gestionar els usuaris i contrasenyes dels equips internament amb la plataforma sense requerir visualització ni coneixement d'aquests paràmetres per part del tècnic en la seva intervenció amb les accions.
10. Usar la pila tecnològica emprada dins l'empresa per a garantir una integració eficient i efectiva.

6. Casos d'ús

Els casos d'ús d'aquest projecte serveixen per definir de manera precisa com els usuaris interactuen amb la plataforma AlphaMonitor en diferents situacions, actuant com a guia per al desenvolupament del sistema, assegurant que compleixi les necessitats dels usuaris. Aquests casos descriuen les noves funcionalitats de la plataforma.

1. **Reiniciar els equips que requereixen d'un reinici de forma automàtica i autònoma:**

    - **Descripció:** Reiniciar el llistat d'equips considerats com a requerits d'un reinici cada cinc minuts.

    - **Actor/s:**

        1. AlphaMonitor – tasca programada al Backend.
        2. Base de dades d'AlphaMonitor.
        3. ApiPulse.
        4. SQS.
        5. OpenSearch.

    - **Condicions prèvies:**

        1. La propietat de reinici automàtic està activa.
        2. L'equip té les seves dades correctament introduïdes a la base de dades.

    - **Flux normal d'execució:**

        1. El mètode programat cada cinc minuts al backend de l'AlphaMonitor recull el llistat d'equips necessaris d'un reinici i consulta la base de dades per obtenir les dades necessàries.

2. La base de dades rep la consulta de l'AlphaMonitor i retorna les dades.

3. L'AlphaMonitor rep les dades i envia el llistat d'equips a reiniciar a l'ApiPulse.

4. L'ApiPulse rep el llistat d'equips a reiniciar i els reinicia, enviant un missatge a la SQS informant que s'ha fet un reinici per cada equip reiniciat.

5. La SQS rep el missatge del reinici, converteix les dades que rep en un missatge que pugui entendre l'OpenSearch i l'envia a l'OpenSearch.

6. L'OpenSearch rep i guarda el missatge.

2. Reiniciar un equip sota demanda:

- **Descripció:** Reiniciar un equip concret sota demanda d'un tècnic SAT.
- **Actor/s:**
  1. Tècnic SAT.
  2. AlphaMonitor.
  3. Base de dades d'AlphaMonitor.
  4. ApiPulse.
  5. SQS.
  6. OpenSearch.
- **Condicions prèvies:**
  1. El tècnic SAT té el nou permís creat per aquestes funcionalitats, necessari per realitzar accions sobre equips remots.

2. L'equip té les seves dades correctament introduïdes a la base de dades.

- **Flux normal d'execució:**

1. El tècnic SAT accedeix a l'equip concret que vol reiniciar i prem sobre el botó d'accions.
2. L'AlphaMonitor mostra les possibilitats d'accions sobre l'equip.
3. El tècnic SAT prem sobre la acció de reiniciar.
4. L'AlphaMonitor recull la petició i la envia a l'ApiPulse, mostra un missatge a mode de feedback informant que la petició de reinici s'ha enviat correctament.
5. L'ApiPulse rep l'equip a reiniciar i el reinicia, enviant un missatge a la SQS informant que s'ha fet un reinici d'aquell equip.
6. La SQS rep el missatge del reinici, converteix les dades que rep en un missatge que pugui entendre l'OpenSearch i l'envia a l'OpenSearch.
7. L'OpenSearch rep i guarda el missatge.

3. Veure els registres d'un equip sota demanda:

- **Descripció:** Veure els registres d'un equip concret sota demanda d'un tècnic SAT.
- **Actor/s:**

1. Tècnic SAT.
2. AlphaMonitor.
3. Base de dades d'AlphaMonitor.
4. ApiPulse.

5. OpenSearch.

- **Condicions prèvies:**

  1. El tècnic SAT té el nou permís creat per aquestes funcionalitats, necessari per realitzar accions sobre equips remots.

  2. L'equip té les seves dades correctament introduïdes a la base de dades.

- **Flux normal d'execució:**

  1. El tècnic SAT accedeix a l'equip concret que del què vol veure'n els registres i prem sobre el botó d'accions.

  2. L'AlphaMonitor mostra les possibilitats d'accions sobre l'equip.

  3. El tècnic SAT prem sobre la acció de veure registres.

  4. L'AlphaMonitor recull la petició i la envia a l'ApiPulse, mostra un missatge a mode de feedback informant que la petició de veure els registres s'ha enviat correctament i obre un mòdul de consola a mode de finestra que mostra el resultat de l'acció.

  5. L'ApiPulse rep l'equip del què mostrar els registres, es connecta a l'equip, demana els registres i els retorna al backend d'AlphaMonitor.

  6. L'AlphaMonitor rep el resultat de l'acció de l'ApiPulse i mostra els registres al mòdul de la consola.

  7. El tècnic SAT visualitza els registres al mòdul de la consola.

4. **Veure la configuració sencera d'un equip sota demanda:**

- **Descripció:** Veure la configuració sencera d'un equip concret sota demanda d'un tècnic SAT.

- **Actor/s:**

  1. Tècnic SAT.

2. AlphaMonitor.
3. Base de dades d'AlphaMonitor.
4. ApiPulse.
5. OpenSearch.

- **Condicions prèvies:**

1. El tècnic SAT té el nou permís creat per aquestes funcionalitats, necessari per realitzar accions sobre equips remots.
2. L'equip té les seves dades correctament introduïdes a la base de dades.

- **Flux normal d'execució:**

1. El tècnic SAT accedeix a l'equip concret que del què vol veure'n els registres i prem sobre el botó d'accions.
2. L'AlphaMonitor mostra les possibilitats d'accions sobre l'equip.
3. El tècnic SAT prem sobre la acció de veure la configuració sencera.
4. L'AlphaMonitor recull la petició i la envia a l'ApiPulse, mostra un missatge a mode de feedback informant que la petició de veure la configuració sencera s'ha enviat correctament i obre un mòdul de consola a mode de finestra que mostra el resultat de l'acció.
5. L'ApiPulse rep l'equip del què mostrar la configuració sencera, es connecta a l'equip, demana la configuració sencera i la retorna al backend d'AlphaMonitor.
6. L'AlphaMonitor rep el resultat de l'acció de l'ApiPulse i mostra la configuració sencera al mòdul de la consola.
7. El tècnic SAT visualitza la configuració sencera al mòdul de la consola.

5. **Recuperar el firmware dels equips disponibles i actualitzar-lo a la base de dades automàticament:**

   - **Descripció:** Recuperar el firmware d'un equip i actualitzar-lo a la base de dades automàticament un cop al mes.

   - **Actor/s:**

     1. AlphaMonitor – tasca programada al Backend.
     2. Base de dades d'AlphaMonitor.
     3. ApiPulse.
     4. SQS.
     5. OpenSearch.

   - **Condicions prèvies:**

     1. La propietat d'actualitzar el firmware automàticament està activa.
     2. L'equip té les seves dades correctament introduïdes a la base de dades.

   - **Flux normal d'execució:**

     1. El mètode programat un cop al mes al backend de l'AlphaMonitor recull el llistat d'equips disponibles dels què actualitzar propietats de firmware i consulta la base de dades per obtenir les dades necessàries.
     2. La base de dades rep la consulta de l'AlphaMonitor i retorna les dades.
     3. L'AlphaMonitor rep les dades i envia el llistat d'equips de què actualitzar les propietats de firmware a l'ApiPulse.
     4. L'ApiPulse rep el llistat d'equips a actualitzar el firmware i els retorna amb les propietats més actuals, enviant un missatge a la SQS informant que s'ha actualitzat el firmware per cada equip consultat.

5. La SQS rep el missatge de l'actualització, converteix les dades que rep en un missatge que pugui entendre l'OpenSearch i l'envia a l'OpenSearch.

6. L'OpenSearch rep i guarda el missatge.

6. **Recuperar el firmware d'un equip i actualitzar-lo a la base de dades sota demanda:**

   - **Descripció:** Recuperar el firmware d'un equip concret i actualitzar-lo a la base de dades sota demanda d'un tècnic SAT.

   - **Actor/s:**

     1. Tècnic SAT.

     2. AlphaMonitor.

     3. Base de dades d'AlphaMonitor.

     4. ApiPulse.

     5. OpenSearch.

   - **Condicions prèvies:**

     1. El tècnic SAT té el nou permís creat per aquestes funcionalitats, necessari per realitzar accions sobre equips remots.

     2. L'equip té les seves dades correctament introduïdes a la base de dades.

   - **Flux normal d'execució:**

     1. El tècnic SAT accedeix a l'equip concret que del què vol recuperar el firmware d'un equip concret i actualitzar-lo a la base de dades i prem sobre el botó d'accions.

     2. L'AlphaMonitor mostra les possibilitats d'accions sobre l'equip.

3. El tècnic SAT prem sobre la acció d'actualitzar firmware.

4. L'AlphaMonitor recull la petició i la envia a l'ApiPulse, mostra un missatge a mode de feedback informant que la petició de actualitzar firmware s'ha enviat correctament i obre un mòdul de consola a mode de finestra que mostra el resultat de l'acció.

5. L'ApiPulse rep l'equip del què recuperar el firmware i actualitzar-lo a la base de dades, es connecta a l'equip, demana les dades del firmware i les retorna al Backend d'AlphaMonitor.

6. L'AlphaMonitor rep el resultat de l'acció de l'ApiPulse i actualitza els camps pertinents de la base de dades.

7. **Actualitzar el fitxer de configuració dels equips disponibles de forma automàtica i autònoma:**

   - **Descripció:** Actualitzar el fitxer de configuració del llistat d'equips disponibles un cop al dia.

   - **Actor/s:**

     1. AlphaMonitor – tasca programada al Backend.

     2. Base de dades d'AlphaMonitor.

     3. ApiPulse.

     4. SQS.

     5. S3.

     6. OpenSearch.

   - **Condicions prèvies:**

     1. La propietat d'actualitzar el fitxer de configuració automàticament està activa.

2. L'equip té les seves dades correctament introduïdes a la base de dades.

- **Flux normal d'execució:**

    1. El mètode programat un cop al dia al backend de l'AlphaMonitor recull el llistat d'equips disponibles dels què actualitzar el fitxer de configuració i envia el llistat a l'ApiPulse.

    2. L'ApiPulse rep el llistat d'equips a actualitzar el fitxer de configuració i els processa de la següent manera:

        1. Es connecta a l'equip i guarda la configuració actual d'aquest a S3 i envia un missatge a la SQS conforme s'ha actualitzat el fitxer actual de l'equip.

        2. Compara aquest arxiu actual amb l'arxiu guardat com a validat a S3, en cas de ser iguals no fa res més però en cas de ser diferents envia un missatge a la SQS conforme els arxius de configuració d'aquell equip no coincideixen.

    3. La SQS rep el missatge que correspon, converteix les dades que rep en un missatge que pugui entendre l'OpenSearch i l'envia a l'OpenSearch.

    4. L'OpenSearch rep i guarda el missatge.

8. Actualitzar el fitxer de configuració d'un equip sota demanda:

- **Descripció:** Actualitzar el fitxer de configuració d'un equip concret sota demanda d'un tècnic SAT.

- **Actor/s:**

    1. Tècnic SAT.

    2. AlphaMonitor.

    3. Base de dades d'AlphaMonitor.

4. ApiPulse.
5. S3.
6. SQS.
7. OpenSearch.

- **Condicions prèvies:**

1. El tècnic SAT té el nou permís creat per aquestes funcionalitats, necessari per realitzar accions sobre equips remots.
2. L'equip té les seves dades correctament introduïdes a la base de dades.

- **Flux d'execució A:**

1. El tècnic SAT accedeix a l'equip concret que del què vol actualitzar el fitxer de configuració i entra a l'apartat de configuració.
2. L'AlphaMonitor mostra la pantalla de configuració i fa una consulta a S3 per recuperar tots els fitxers de configuració d'aquell equip.
3. S3 retorna el llistat de fitxers disponibles de l'equip a AlphaMonitor.
4. AlphaMonitor mostra els fitxers obtinguts amb el resultat de la comparativa dels continguts dels fitxers si ha estat possible d'efectuar.
5. El tècnic SAT valida el fitxer més actual de l'equip.
6. L'AlphaMonitor recull la petició de validar el fitxer i la envia a l'ApiPulse, mostra un missatge a mode de feedback informant que la petició de validar el fitxer de configuració s'ha enviat correctament.
7. L'ApiPulse rep la petició, envia a S3 i a OpenSearch.
8. S3 rep i emmagatzema les dades.

9. OpenSearch rep i emmagatzema les dades.

- **Flux d'execució B:**

1. El tècnic SAT accedeix a l'equip concret que del què vol actualitzar el fitxer de configuració i entra a l'apartat de configuració.
2. L'AlphaMonitor mostra la pantalla de configuració i fa una consulta a S3 per recuperar tots els fitxers de configuració d'aquell equip.
3. S3 retorna el llistat de fitxers disponibles de l'equip a AlphaMonitor.
4. AlphaMonitor mostra els fitxers obtinguts amb el resultat de la comparativa dels continguts dels fitxers si ha estat possible d'efectuar.
5. El tècnic SAT puja un fitxer local del seu sistema com a fitxer de configuració de l'equip.
6. L'AlphaMonitor recull la petició de pujar el fitxer i l'envia a l'ApiPulse.
7. L'ApiPulse rep la petició, envia a S3 i a OpenSearch.
8. S3 rep i emmagatzema les dades.
9. OpenSearch rep i emmagatzema les dades.

9. Reconeixement i gestió del dispositiu remot:

- **Descripció:** Reconèixer el dispositiu remot connectat i gestionar el tractament de les seves dades, incloent-hi l'API i el port.
- **Actors:**

1. AlphaMonitor.
2. Base de dades d'AlphaMonitor.

- **Condicions prèvies:**

1. El dispositiu remot es troba a la base de dades amb les seves dades necessàries per realitzar una connexió correctes, que són usuari, contrasenya, el port i la referencia al model que li permet saber la seva API.

- **Flux normal d'execució:**

1. El sistema avalua que s'ha de realitzar una acció sobre un dispositiu remot.

2. Recupera les dades de connexió de la base de dades, com l'API, el port, l'usuari i la contrasenya del dispositiu remot.

3. Gestiona les dades del dispositiu remot segons les especificacions definides.

10. Recollida, anàlisi i tractament del monitoratge d'AlphaMonitor sobre l'estat del dispositiu:

- **Descripció:** Recollir, analitzar i tractar el resultat de monitoritzar l'estat del dispositiu remot per part d'AlphaMonitor.

- **Actors:**

1. AlphaMonitor.

- **Condicions prèvies:**

1. La propietat de monitoritzar està activa.

- **Flux normal d'execució:**

1. El sistema monitoritza l'estat del dispositiu remot.

2. Processa les dades d'anàlisi per a l'ús posterior, com la presa de decisions o la generació de notificacions.

11. Avaluació automàtica d'accions per l'AlphaMonitor:

- **Descripció:** Avaluar de manera automàtica si l'AlphaMonitor ha de realitzar alguna acció, com ara un reinici, sobre el dispositiu remot.
- **Actors:**
    1. AlphaMonitor.
    2. ApiPulse.
- **Condicions prèvies:**
    1. El sistema ha detectat mitjançant el sistema de monitoratge un error o un problema amb el dispositiu remot.
    2. El dispositiu passa els filtres per ser un equip sobre el qual es poden realitzar accions en remot.
- **Flux normal d'execució:**
    1. El sistema de monitoratge de la plataforma AlphaMonitor detecta un error en el dispositiu remot i avalua automàticament si cal realitzar alguna acció per solucionar l'error. AlphaMonitor envia una comanda a l'ApiPulse per realitzar l'acció requerida.
    2. ApiPulse rep la comanda i efectua l'acció pertinent.

12. Enviament de dades d'execucions a la SQS i a OpenSearch:

- **Descripció:** Enviar les dades necessàries de les execucions per part de l'ApiPulse a OpenSearch.
- **Actors:**
    1. ApiPulse.
    2. SQS.
    3. OpenSearch.

- **Condicions prèvies**

    1. Les dades d'execució estan disponibles i cal enviar-les a OpenSearch.

- **Flux normal d'execució:**

    1. ApiPulse finalitza la acció i envia a la SQS.
    2. SQS rep les dades, les transforma en dades òptimes pel correcte enteniment a OpenSearch i envia a OpenSearch.
    3. OpenSearch rep les dades, on queden emmagatzemades.

13. Visualitzar l'històric de les accions realitzades sobre equips:

- **Descripció:** Visualitzar l'històric de les accions realitzades sobre els equips remots en un apartat del Frontend anomenat *Quality control* de l'AlphaMonitor en forma de llistat, on a cada fila del llistat s'inclou informació tal com:

    1. Tipus d'acció.
    2. Data de realització de l'acció.
    3. Autor de l'acció, ja sigui el mateix sistema o un tècnic SAT.

- **Actors:**

    1. Tècnic SAT.
    2. AlphaMonitor.
    3. OpenSearch.

- **Condicions prèvies:**

    1. El tècnic SAT té el nou permís creat per aquestes funcionalitats, necessari per realitzar accions sobre equips remots.

- **Flux normal d'execució:**

1. El tècnic SAT accedeix a l'apartat de *Quality control* a la barra lateral esquerra de l'AlphaMonitor.
2. L'AlphaMonitor ataca la base de dades d'OpenSearch.
3. OpenSearch rep la petició i retorna el resultat a l'AlphaMonitor.
4. L'AlphaMonitor rep el resultat i el representa en forma de llista paginada.
5. El tècnic SAT visualitza l'històric on cada tipus d'acció és representada amb un color diferent, amb la possibilitat de navegar a l'equip concret si prem sobre la fila d'aquest al llistat.

14. Visualitzar el gràfic de les accions realitzades sobre un equip:

- **Descripció:** Visualitzar el gràfic de les accions realitzades sobre els un equip remot en l'apartat del dispositiu anomenat gràfics de l'AlphaMonitor.
- **Actors:**
  1. Tècnic SAT.
  2. AlphaMonitor.
  3. OpenSearch.
- **Condicions prèvies:**
  1. El tècnic SAT té el nou permís creat per aquestes funcionalitats, necessari per realitzar accions sobre equips remots.
- **Flux normal d'execució:**
  1. El tècnic SAT accedeix a l'apartat de gràfics de l'equip del que vol fer la consulta.
  2. L'AlphaMonitor ataca la base de dades d'OpenSearch.
  3. OpenSearch rep la petició i retorna el resultat a l'AlphaMonitor.

4. L'AlphaMonitor rep el resultat i el representa en forma de gràfic de barres, cada tipus d'acció representada amb un color diferent explicat a la llegenda del gràfic, amb la possibilitat situar-se sobre cada barra i veure les dades de l'acció tals com:

    1. Tipus d'acció.

    2. Data de realització de l'acció.

    3. Autor de l'acció, ja sigui el mateix sistema o un tècnic SAT.

5. El tècnic SAT visualitza el gràfic amb les dades correctament representades.

7. Anàlisi i disseny

Abans de començar a implementar les noves funcionalitats, s'observa l'estat actual de la base de dades així com de la plataforma AlphaMonitor, i s'arriba a la conclusió de la necessitat de reestructurar la base de dades abans de fer la implementació dels nous camps per les noves funcionalitats del projecte, i es dissenyen aquests canvis prioritzant:

- Eficiència i eficàcia en l'obtenció de dades.
- Mantenibilitat i llegibilitat tant a nivell de base de dades com a nivell de codi.

7.1 Reestructuració de la base de dades

L'entitat Equipment és l'entitat pare dels tipus d'equipament que hi ha, que són els següents:

- Device
- Communication
- Supply

Per tant, aquestes tres entitats hereten de l'entitat Equipment, de manera que cada *device*, cada *communication* i cada *supply* tenen a base de dades una fila a la taula Equipment i una fila a la taula de l'entitat que siguin.

Amb motiu de corregir inconsistències a la base de dades, es lliguen els camps de *model*, *type* i *brand* de forma que no siguin camps de text lliures als formularis de creació i edició de cada dispositiu i no puguin donar peu a equivocacions per part dels tècnics.

El resultat d'això és eliminar els camps *model*, *type* i *brand* de la taula d'Equipment i crear una nova columna anomenada model_id, que té la referència com una clau forana a una nova taula anomenada Model.

Column Name	#	Tipo de datos	Identidad	Collation	No Nulo	Por defecto	Comentario
id	1	serial4			[v]	nextval('equipment_id_seq'::regclass)	
parent_id	2	int4			[ ]		
equipment_type	3	public."equipment_type			[v]		
id_monitor	4	varchar(150)		default	[ ]		
name	5	varchar(150)		default	[ ]		
description	6	varchar(2000)		default	[ ]		
date	7	date			[ ]		
stage	8	public."stage"			[v]		
ticket_id	9	varchar(50)		default	[ ]		
ticket_description	10	varchar(250)		default	[ ]		
tags	11	jsonb			[v]		
id_system	12	int4			[v]		
created_by	13	varchar(100)		default	[ ]		
created_on	14	timestamp(3)			[ ]		
updated_by	15	varchar(100)		default	[ ]		
updated_on	16	timestamp(3)			[ ]		
status	17	public."status"			[v]		
contract	18	public."contract"			[v]	'CONTRACT'::contract	
maintenance	19	public."maintenance"			[v]		
additional_fields	20	jsonb			[ ]	'{}'::jsonb	
serial	21	varchar(150)		default	[ ]		
phase	22	int4			[ ]		
port	23	int4			[ ]		
model_id	24	int4			[v]	1	
username	25	varchar(50)		default	[ ]		
password	26	varchar(50)		default	[ ]		
warranty_expiration_date	27	date			[ ]		
warranty_duration	28	varchar(10)		default	[ ]		

Figura 5. Resultat de l'entitat de la base de dades Equipment. Font: Elaboració pròpia.

Column Name	#	Tipo de datos	Identidad	Collation	No Nulo	Por defecto	Comentario
id	1	serial4			[v]	nextval('model_id_seq'::regclass)	
name	2	varchar(100)		default	[ ]		
type	3	varchar(100)		default	[ ]		
brand_id	4	int4			[ ]		
created_by	5	varchar(100)		default	[ ]		
created_on	6	timestamp(3)			[ ]		
updated_by	7	varchar(100)		default	[ ]		
updated_on	8	timestamp(3)			[ ]		
status	9	public."status"			[v]	'ACTIVE'::status	
api	10	varchar(100)		default	[ ]	'NONE'::character varying	

Figura 6. Nova entitat de la base de dades Model. Font: Elaboració pròpia.

Column Name	#	Tipo de datos	Identidad	Collation	No Nulo	Por defecto	Comentario
id	1	serial4			[v]	nextval('brand_id_seq'::regclass)	
name	2	varchar(50)		default	[ ]		

Figura 7. Nova entitat de la base de dades Brand. Font: Elaboració pròpia.

D'aquesta mateixa manera, a la taula Model hi ha un camp brand_id amb la referència com a clau forana a la nova taula Brand.

Per tal d'implementar les noves funcionalitats s'introdueix un nou camp a les taules d'Equipment i Model, necessaris per possibilitar les connexions amb les APIs de cada equip, que són els següents:

- Taula Equipment disposa d'un nou camp anomenat *port.*
- Taula Model disposa d'un nou camp anomenat *API.*

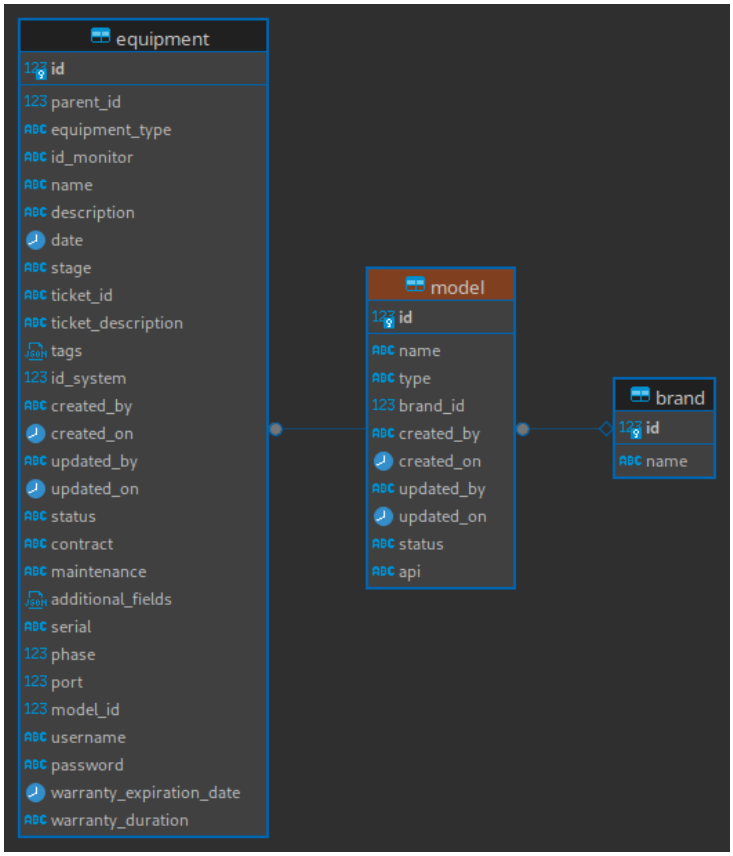

Figura 8. Esquema relacional resultant de les entitats Equipment, Model i Brand. Font: Elaboració pròpia.

S'ha completat una renovació exhaustiva de la infraestructura de la base de dades, englobant tant les taules com les seves interconnexions. La versió actualitzada de la base de dades ha estat dissenyada amb la finalitat d'arreglar inconsistències existents i prevenir inconsistències futures, amb l'objectiu de promoure l'adopció de metodologies òptimes i assegurar una organització transparent de les dades.

L'actualització de la base de dades representa una intervenció crucial que abordava una problemàtica persistent en la gestió dels equips per part del personal tècnic.

Aquesta implementació ha solucionat de manera efectiva diversos obstacles en la gestió quotidiana dels equips, així com permès implementar les noves funcionalitats d'aquest projecte.

8. Desenvolupament

8.1 Implementació de les funcionalitats a AlphaMonitor

En relació a la implementació de les noves funcionalitats a AlphaMonitor, es poden diferenciar entre les accions que es realitzen sota demanda a un equip en concret per part d'un tècnic de l'equip SAT i les que es realitzen de manera automàtica i autònoma per part de la mateixa plataforma quan aquesta així ho decideix.

Donat que certes accions són accions de rellevant importància en el funcionament dels equips, es consideren dos factors a integrar:

- Hi ha accions que han de quedar auditades, per tant es guarden a l'OpenSearch i poden ser consultades a un apartat del Frontend d'AlphaMonitor anomenat *Quality control.* Es guarden dades tals com el tipus d'acció, la data de realització de l'acció i el nom del tècnic que les ha dut a terme en cas que siguin realitzades sota demanda, i en cas que les hagi fet la pròpia plataforma queden auditades amb el nom d'aquesta.
- No tots els tècnics SAT han de poder realitzar accions sobre equips remots, per tant es crea un permís especial que pot ser atribuït a usuaris de l'equip de tècnics SAT que permeti realitzar aquestes accions només quan es disposa d'aquest permís.

Les accions a guardar l'històric amb el nom de l'autor, la data i el tipus d'acció són:

- Reinici.
- Actualitzar el fitxer de configuració actual.
- Validar el fitxer de configuració.

Aquestes accions en el moment de fer-se ja sigui per part d'un tècnic o per part de la mateixa plataforma, s'envien a la SQS de manera que acaben guardant-se a l'OpenSearch, base de dades de la qual se'n extreuen les dades per tal que posteriorment aquestes siguin auditades en forma de taula al Frontend de l'AlphaMonitor així com representades en un gràfic a l'apartat gràfics de l'equip.

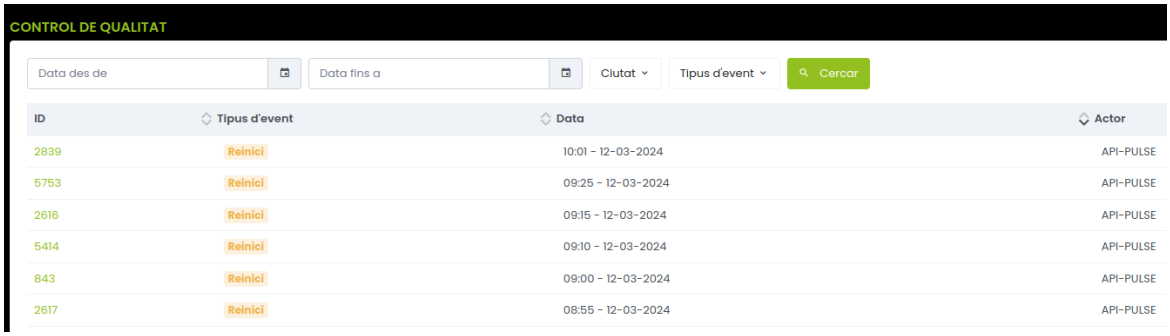

Figura 9. Apartat Quality control al Frontend d'AlphaMonitor, amb dades d'autorestarts. Font: Elaboració pròpia.

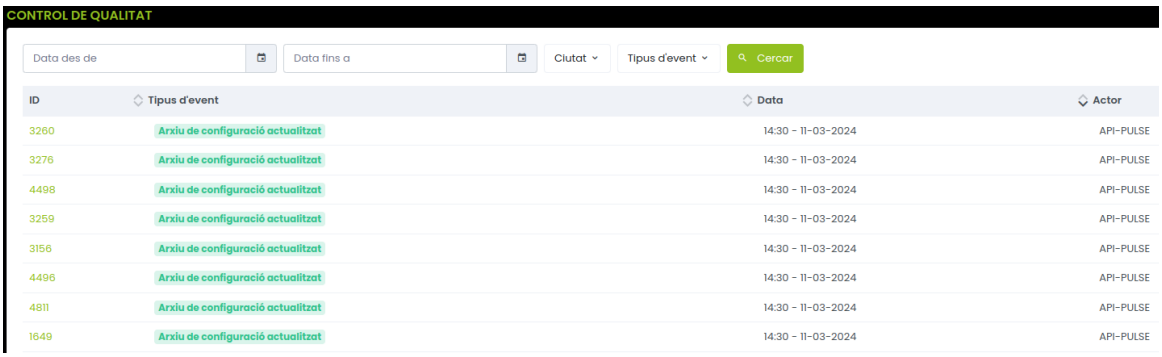

Figura 10. Apartat Quality control al Frontend d'AlphaMonitor, amb dades d'autoconfiguracions. Font: Elaboració pròpia.

Figura 11. Apartat Gràfics al Frontend d'AlphaMonitor, amb dades d'autorestarts. Font: Elaboració pròpia.

Figura 12. Apartat Gràfics al Frontend d'AlphaMonitor, amb dades d'autoconfiguracions. Font: Elaboració pròpia.

8.1.1 Accions sota demanda

Per realitzar les accions sota demanda, s'ha afegit un botó a cada tipus d'equipament en forma de desplegable que mostra les accions possibles a realitzar sobre l'equip. També s'han afegit noves dades a la fitxa tècnica de l'equip com el port, i s'han estandarditzat les dades de *model, type* i *brand.*

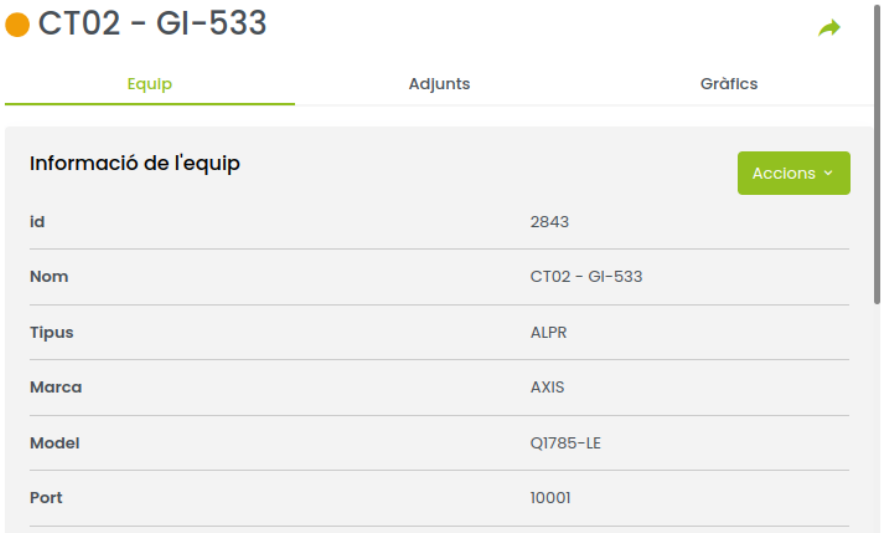

Figura 13. Botó d'accions afegit a cada tipus d'equipament, en aquest cas d'un device. Font: Elaboració pròpia.

Figura 14. Botó d'accions desplegat que mostra les possibles accions a realitzar. Font: Elaboració pròpia.

S'ha creat un mòdul de consola que imprimeix el resultat de la acció en forma de missatge en cas que la acció sigui per visualitzar dades de l'equip.

Aquest mòdul de consola ha estat creat amb la finalitat de mostrar el missatge retornat per l'equip d'una manera més entenedora i usable, donat que conté eines d'ajuda a la interacció dels tècnics SAT tals com:

- Modes diferents de visualització del contingut:
  - Mode clar.
  - Mode fosc.
- Desplaçar-se directament fins al final del contingut.
- Barra lateral per desplaçar-se progressivament en el contingut.
- Actualitzar el contingut sota demanda (refrescar).
- Actualitzar automàticament el contingut quan es detecten canvis.
- Tancar el mòdul de consola.

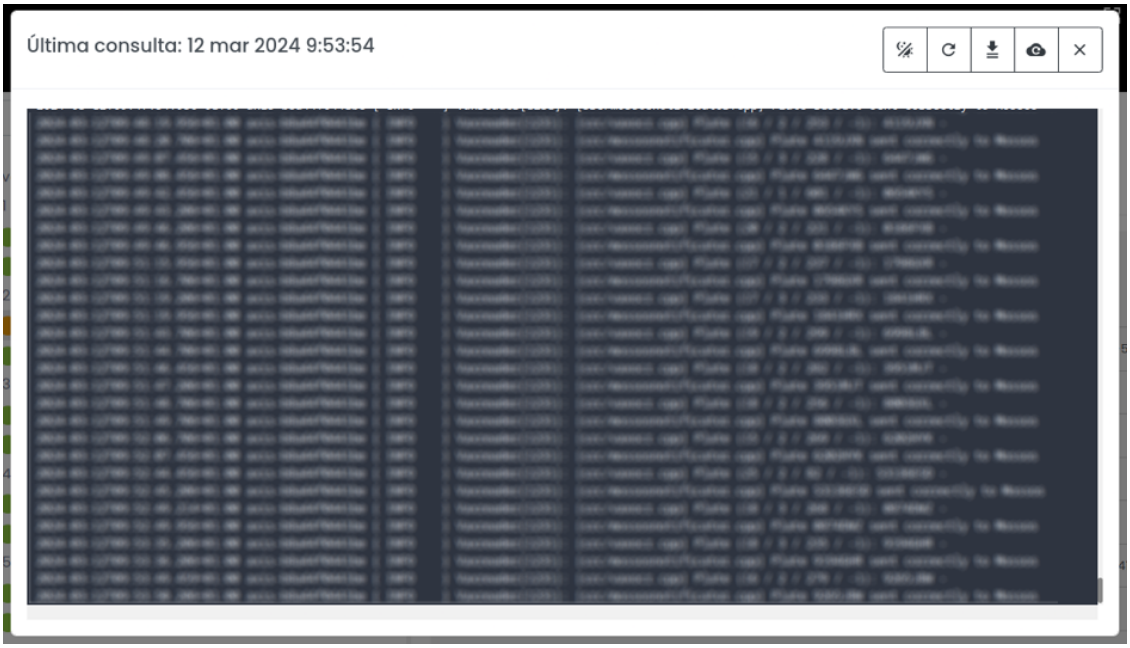

Figura 15. Mòdul de consola implementat en les accions que han de retornar un missatge (difuminat afegit per no mostrar dades reals). Font: Elaboració pròpia.

A mode d'incrementar la usabilitat i produir un software el més entenedor possible pels usuaris finals que són els tècnics SAT, s'ha afegit un requadre verd a mode de *feedback* que indica quan la petició ha estat correctament enviada i quan ha fallat. Per aconseguir aquesta millora es fa ús d'una classe d'objecte que entenen tant l'AlphaMonitor com l'ApiPulse amb

el que s'envien els missatges. Un dels atributs de l'objecte és un codi d'execució, on es guarda el codi de la crida http resultat de l'acció.

```
public class ExecutionResponseDto {

    3 usages
    private Integer id;
    3 usages
    private String output;
    3 usages
    private Integer exitCode;
    3 usages
    private String error;
```

Figura 16. Classe ExecutionResponseDto usada per la comunicació d'execucions resultats de crides http i l'enviament de missatges entre AlphaMonitor i ApiPulse.

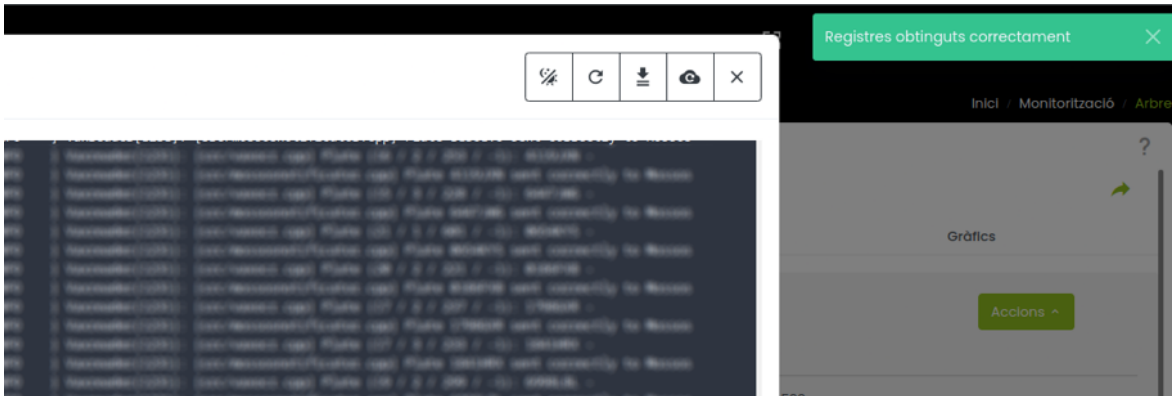

Figura 17. Addició de feedback a mode de missatge en requadre superior dret, cas d'èxit. Font: Elaboració pròpia.

Figura 18. Addició de feedback a mode de missatge en requadre superior dret, cas d'error. Font: Elaboració pròpia.

En el cas de la autoconfiguració sota demanda, les accions són disponibles a l'apartat de configuració de cada equip, amb les possibilitats esmentades a continuació:

- Comprovar si el fitxer de configuració de l'equip ha canviat respecte de l'últim que es tenia com a vàlid.
- Visualitzar el contingut de cadascun dels fitxers, tant de l'actual de l'equip com del guardat com a vàlid.
- Validar el fitxer actual de la càmera.
- Pujar un nou fitxer de configuració que automàticament es considera el fitxer validat.

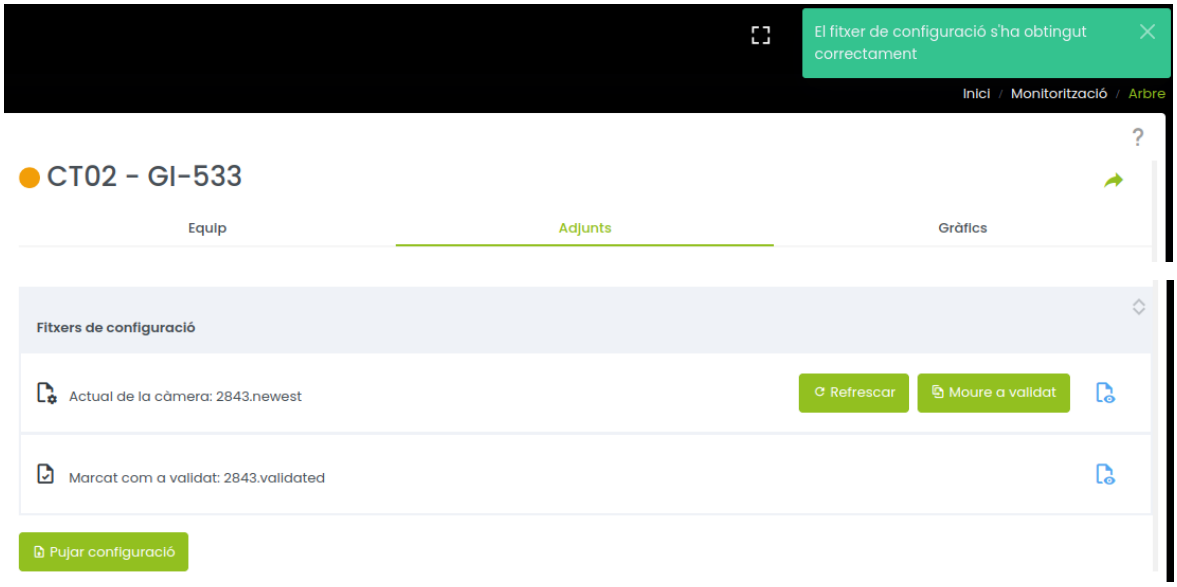

Figura 19. Actualització del fitxer de configuració a l'apartat de configuració d'un equipament al Frontend d'AlphaMonitor, cas on els fitxers són iguals i el fitxer de configuració no ha canviat. Font: Elaboració pròpia.

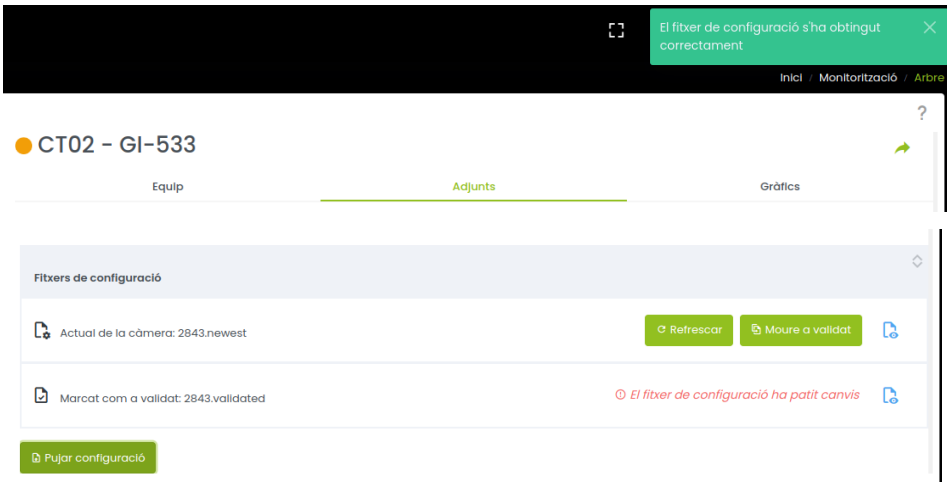

Figura 20. Actualització del fitxer de configuració a l'apartat de configuració d'un equipament al Frontend d'AlphaMonitor, cas on els fitxers són diferents i el fitxer de configuració ha canviat. Font: Elaboració pròpia.

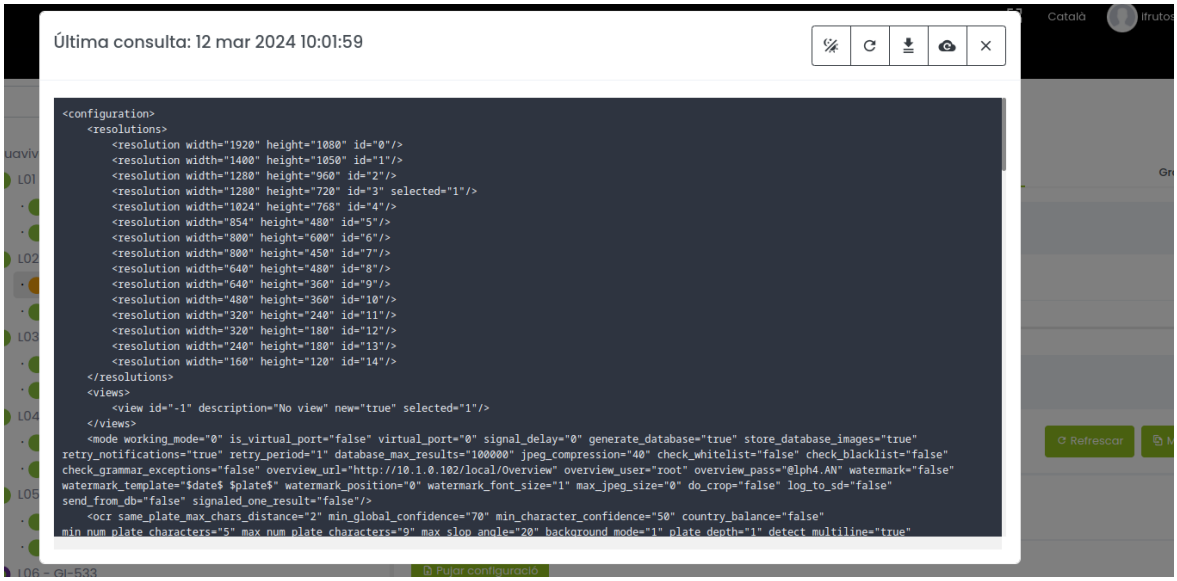

Figura 21. Visualització del contingut del fitxer de configuració. Font: Elaboració pròpia.

8.1.2 Accions automàtiques

AlphaMonitor usa les dades obtingudes pel seu monitoratge dels equips i avalua si ha de realitzar accions sobre l'equip o no.

Les accions automàtiques a realitzar són les següents:

- Reinici automàtic.
- Actualització del fitxer de configuració automàtica.
- Actualització de les dades del firmware automàtica.
- Actualització de credencials automàtica.

Donat que són accions realitzades per la pròpia plataforma, no tenen cap apartat visual a l'AlphaMonitor, però queden auditades d'igual manera que si es fan sota demanda per part d'un tècnic SAT.

8.2 Flux i funcionament

8.2.1 AlphaMonitor

Des del Frontend, el tècnic SAT accedeix a l'equip concret sobre el que vol realitzar una acció, com per exemple reiniciar un *device*, prem sobre el botó d'accions i clica sobre l'acció que desitja fer.

L'AlphaMonitor implementa una solució òptima per recollir el tipus de fabricant i API de l'equip gràcies a un patró de software anomenat patró estratègia. AM recull la petició en un objecte del tipus que sigui el fabricant, que implementa la interfície d'APIs que implementen tots els fabricants, per exemple, d'un equip amb model Axis el tipus d'objecte és AxisApi i implementa la interfície Api. Aquesta implementació utilitza el patró de disseny Estratègia.

El mètode getApi(ApiEnum apiEnum) decideix quina implementació de EquipmentApi usar basant-se en el valor de ApiEnum. Si el valor es VAPIX (llibreria del fabricant Axis), s'usa AxisApi, en qualsevol altre cas, es fa servir NoneApi.

Per altra banda, AxisApi i NoneApi implementen la interfície EquipmentApi, la qual cosa és un exemple del patró de disseny Polimorfisme. Ambdues classes proporcionen la seva pròpia implementació dels mètodes definits en la interfície EquipmentApi.

El patró de disseny Estratègia proporciona diverses avantatges a la classe ApiManagerServiceImpl:

- Flexibilitat: El patró Estratègia permet a la classe ApiManagerServiceImpl canviar entre diferents implementacions d'API (AxisApi i NoneApi) en temps d'execució. Això es fa a través del mètode getApi(), que retorna una instància d'EquipmentApi basada en el valor de ApiEnum. Aquesta flexibilitat facilita l'addició de noves implementacions d'API en el futur sense modificar la classe ApiManagerServiceImpl.
- Desacoblament: La classe ApiManagerServiceImpl està desacoblada de les implementacions específiques de l'API. Interactua amb una API a través de la interfície EquipmentApi, no directament amb les classes AxisApi o NoneApi.

Aquest desacoblament promou la modularitat del codi i la separació de responsabilitats.

- Testabilitat: El patró Estratègia millora la capacitat de prova de la classe ApiManagerServiceImpl. Es pot simular fàcilment la interfície EquipmentApi durant les proves unitàries, el que permet provar la classe ApiManagerServiceImpl de forma aïllada de les seves dependències.
- Reutilització de codi: Les classes AxisApi i NoneApi poden ser reutilitzades en diferents contextos. Qualsevol classe que necessiti interactuar amb una API pot fer servir aquestes classes, sempre que ho faci a través de la interfície EquipmentApi.
- Consistència: Totes les operacions de l'API (com ara *restart*, *showSettings*, *showLogs*, etc.) s'accedeixen de manera consistent a través de la interfície EquipmentApi, independentment de la implementació específica de l'API. Aquesta consistència fa que el codi sigui més fàcil d'entendre i mantenir.

En el moment que es construeix l'objecte, es posen les dades necessàries per dur a terme l'acció. Al ser crides http, les dades essencials són l'API, el port, l'usuari, la contrasenya i la acció a fer, que és part de la URL final.

Un cop es té l'objecte s'envia a l'ApiPulse, que en cas d'estar operatiu agafa la petició i mostra un missatge a mode de feedback informant que la petició de l'acció, per exemple de reinici, s'ha enviat correctament.

8.2.2 ApiPulse

Per rebre peticions a l'ApiPulse s'ha creat un objecte genèric del que hereten totes les peticions que es faran dels diferents fabricants amb diferents APIs. Aquest objecte es diu ApiRequest, i estenen d'ell tots els tipus de peticions dels diferents fabricants, per exemple, l'objecte AxisRequest estén d'ApiRequest i afegeix camps específics de la seva implementació.

ApiPulse disposa de dos *endpoints*:

1. *Endpoint* processar missatges de tipus ApiRequest.
2. *Endpoint* per processar llistes de missatges de tipus ApiRequest.

L'*endpoint* per processar missatges individuals és el que s'usa per accions sota demanda i enviar peticions individuals d'equips concrets, mentre que l'endpoint per processar llistes és el que s'usa per accions automàtiques.

ApiPulse implementa una solució relacionada amb paral·lelisme i concurrència, ja que el mode de processar missatges usa *threads* o fils d'execució, que van consumint missatges de la cua mentre aquesta no sigui buida.

La classe ActionServiceImpl és una implementació de la interfície ActionService. Aquesta classe s'utilitza per processar missatges de tipus ApiRequestDto.

La classe ActionServiceImpl té les següents característiques:

- Està anotada amb @Service, la qual cosa indica que és una classe de servei en el context de Spring Boot. Això significa que Spring crea una instància d'aquesta classe i la gestiona com a *bean* dins del contenidor de Spring.
- Té una llista d'accions (List<Action> actions) i un Executor threadPoolTaskExecutor. Les accions són les tasques que es poden dur a terme i l'Executor és l'encarregat d'executar aquestes tasques en diferents fils.
- Té un mètode processMessage que accepta una llista de ApiRequestDto. Aquest mètode itera sobre cada ApiRequestDto i executa el mètode processMessage per a cadascun en un fil separat.
- Té un altre mètode processMessage que accepta un sol ApiRequestDto. Aquest mètode itera sobre totes les accions disponibles i si troba una acció que coincideix amb l'acció i l'ApiEnum del ApiRequestDto, processa el missatge amb aquesta acció. Si no troba cap acció coincident, retorna una resposta buida (VOID_RESPONSE).

La raó per la qual és beneficiós dissenyar la classe d'aquesta manera és la separació de responsabilitats i la concurrència.

- Separació de responsabilitats: Cada acció és una classe separada que implementa la interfície Action. Això significa que cada acció té la seva pròpia classe que s'encarrega de la seva lògica de processament. Això fa que el codi sigui més fàcil de mantenir i provar.

- Concurrència: En utilitzar un Executor per processar cada missatge en un fil separat, es poden processar múltiples missatges alhora. Això pot millorar el rendiment de l'aplicació, especialment si el processament d'un missatge és una operació que consumeix molt temps.

La interfície Action defineix un contracte per a les classes que la implementen. Aquest contracte inclou dos mètodes:

1. boolean isAction(ActionEnum action, ApiEnum api): Aquest mètode pren dos paràmetres, un ActionEnum i un ApiEnum, i retorna un booleà. Les classes que implementen aquesta interfície han de proporcionar una implementació per a aquest mètode que determini si l'acció és compatible amb l'API.
2. ExecutionResponseDto process(ApiRequestDto dto): Aquest mètode pren un ApiRequestDto com a paràmetre i retorna un ExecutionResponseDto. Les classes que implementen aquesta interfície han de proporcionar una implementació per a aquest mètode que processi el ApiRequestDto i retorni un ExecutionResponseDto.

Les classes que implementen la interfície Action han de proporcionar implementacions per aquests dos mètodes. Aquestes classes representen diferents accions que es poden dur a terme i cadascuna d'elles ha de saber si pot gestionar una acció específica i com processar-la.

Per exemple, si es té una classe ActionA que implementa la interfície Action, aquesta classe ha de proporcionar una implementació per a isAction que retorni true si l'acció és compatible amb ActionA i false en cas contrari. A més, ha de proporcionar una implementació per a process que processi el ApiRequestDto d'una manera específica per a ActionA.

En el codi de ActionServiceImpl, s'utilitzen aquestes classes que implementen la interfície Action. En el mètode processMessage(ApiRequestDto message), s'itera sobre totes les accions disponibles i es crida al mètode isAction de cada acció. Si isAction retorna true, es crida al mètode process d'aquella acció per processar el missatge.

Això permet que el codi sigui flexible i extensible. Si en el futur es necessita afegir una nova acció, simplement es pot crear una nova classe que implementi la interfície Action i es pot afegir a la llista d'accions a ActionServiceImpl. No és necessari modificar el codi existent a ActionServiceImpl.

Quan l'ApiPulse rep la comanda, un *thread* o fil d'execució l'executa, per exemple l'equip a reiniciar i el reinicia, envia un missatge a la SQS informant que s'ha fet un reinici d'aquell equip.

La SQS rep el missatge del reinici i allà espera a ser consumit.

Quan l'AM Consumer pot, agafa el missatge de la cua, converteix les dades que conté en un missatge amb el format que pugui entendre l'OpenSearch i l'envia a l'OpenSearch.

L'OpenSearch rep i guarda el missatge que s'usa posteriorment per obtenir gràfics d'aquell equip com per mostrar a la taula de *quality control* amb la resta d'accions sobre equips.

Per tal d'optimitzar al màxim el temps de resposta, s'ha implementat una cua a l'ApiPulse on entren tots els missatges. Els missatges en forma de llista, com són resultats de processos interns, s'afegeixen a la cua, mentre que els missatges individuals passen per davant de la cua, ja que es vol que si un tècnic SAT sol·licita una acció a l'AlphaMonitor, l'ApiPulse pugui executar-la el més ràpid possible, fent esperar els processos d'equips automàtics, ja que aquests no estan sent esperats per ningú.

Per les accions automàtiques s'han desenvolupat mètodes amb notació de Spring "@Scheduled" per ser executats cada cert temps, depenent del requeriment de cada acció automàtica:

- En cas dels reinicis automàtics, el mètode s'executa cada cinc minuts.
- En cas de l'actualització del fitxer de configuració automàtica, el mètode s'executa un cop al dia.
- En cas de l'actualització de les dades de firmware automàtic, el mètode s'executa un cop al mes.

Les accions s'envien en forma de llistat d'equips i es reben a l'ApiPulse en format d'objectes genèrics ApiRequest, que són reconvertits de nou en objectes del fabricant, per exemple AxisRequest.

Els *threads* van consumint els missatges de la cua mentre aquesta encara tingui missatges i les accions són executades, reduint el nombre de missatges de la cua a cada acció que es realitza sobre un equip.

8.3 Documentació

El nou microservei producte d'aquest treball de final de grau precisa de la documentació adequada per al seu correcte desenvolupament, usabilitat i manteniment. És per això que s'ha realitzat una part del treball d'enginyeria trivial, creant les corresponents documentacions per a l'empresa.

1. Documentació a GitLab (README.md del projecte):

- Aquesta documentació és més formal i a nivell global del projecte.
- Està destinada a un públic més ampli, incloent-hi *stakeholders* i altres membres de l'organització que necessitin entendre els objectius generals i l'arquitectura del projecte.

2. Documentació tècnica amb Obsidian [14]:

- Destinada específicament als desenvolupadors o futurs mantenidors del projecte, al departament de R+D+I.
- Aquesta documentació detalla el funcionament del codi, la lògica interna del microservei, i proporciona guies per a l'extensió i la manutenció del mateix.
- Inclou instruccions detallades sobre la configuració de l'entorn de desenvolupament, la integració amb altres serveis i la resolució de problemes comuns.

Aquestes documentacions escrites en llenguatge markdown [15] asseguren que tant els actuals com els futurs desenvolupadors puguin treballar eficientment amb el microservei, mantenint una alta qualitat i continuïtat en el desenvolupament del projecte.

8.4 Modificacions en els requeriments

A l'inici del projecte es van establir uns objectius, i per aconseguir-los es divideix el projecte en diferents tasques que s'han anat assolint de manera satisfactòria, però la realitat és que s'ha descartat una de les tasques i s'han afegit d'altres.

8.4.1 Requeriments no realitzats

- Gestor de contrasenyes: Durant el transcurs d'altres tasques amb preferència abans del gestor de contrasenyes, l'equip de tècnics SAT descobreix una manera d'assolir l'objectiu d'aquesta per una altra banda. Amb aquesta nova metodologia i sabent que els enviaments de dades de l'AlphaMonitor amb el microservei extern són amb validació d'autentificació mitjançant un *token*, conjuntament amb el cap del departament tècnic es conclou que aquesta tasca s'assoleix per aquesta via alternativa.

8.4.2 Requeriments afegits

- Interfície de cerca avançada: Aquesta nova interfície ajuda els tècnics a cercar informació de manera més ràpida i eficient, millorant significativament la seva capacitat de resposta.
- Interfície de control de qualitat: Aquesta interfície permet als tècnics dur a terme tasques de control de qualitat de manera més àgil, garantint així un millor compliment dels estàndards de qualitat establerts.

Les modificacions mencionades s'implementen per assegurar una major eficiència i eficàcia en el treball dels tècnics, contribuint de manera positiva a l'objectiu global del projecte i encaixant les visions i formes de treballar del departament de R+D+I amb les del departament tècnic.

8.5 Possibles ampliacions futures

Després de l'èxit inicial del projecte, s'han identificat diverses àrees on es poden realitzar ampliacions per millorar encara més la funcionalitat i eficàcia del sistema:

- Implementació d'intel·ligència artificial per a la diagnosi predictiva:
    - La integració de sistemes de *machine learning* per analitzar les dades històriques dels equips i predir possibles fallades abans que es produeixin.

Això permet una intervenció proactiva per part dels tècnics, reduint els temps d'inactivitat dels equips i millorant la satisfacció dels clients.

- Integració amb Nous Dispositius i Tecnologies:
  - L'ampliació de la compatibilitat del sistema per incloure nous dispositius i tecnologies emergents utilitzades pels tècnics. Això assegura que el sistema es mantingui al dia amb els avenços tecnològics i suporti una gamma més àmplia d'equips.
  - Integració de nous fabricants de manera fàcil i ràpida per part dels desenvolupadors de l'equip de R+D+I, ja que s'han emprat patrons i pràctiques d'enginyeria del software que així ho permeten.
- Desenvolupament d'Aplicacions Mòbils:
  - La creació d'aplicacions mòbils que permetin als tècnics accedir a totes les funcionalitats del sistema des de qualsevol lloc, proporcionant major flexibilitat i capacitat de resposta davant incidències. És un punt a tenir en compte sobretot pels tècnics de camp que es desplacen a cada punt on hi ha equipament d'Alphanet instal·lat.

Aquestes ampliacions possibles reflecteixen el compromís continu de millorar el sistema i assegurar que segueix responent a les necessitats canviants dels tècnics i clients de manera eficient i efectiva. Tenen en compte tecnologies emergents que són ja eines potents i presents en el moment de la realització d'aquest treball de final de grau com ara la intel·ligència artificial, aplicacions mòbils i el machine learning.

8.6 *Testing*

El *testing* d'aquest treball de final de grau s'ha realitzat de manera progressiva i contínua, donat que les tasques s'han lliurat i desplegat a l'entorn de producció mitjançant s'han anat assolint, i els clients (tècnics SAT) han fet ús diari de les noves implementacions, donant *feedback* continuat, entrant com a tasques de canvis i millores en els nous *sprints*.

9. Conclusions

El treball de final de grau ha aconseguit complir els objectius plantejats inicialment i ha satisfet els requeriments establerts. S'han aconseguit implementar solucions que permeten la resolució automàtica d'errors en equips remots mitjançant una plataforma que ha evolucionat de la plataforma que era abans i ha pres un rol de sistema decisional autònom.

La plataforma amb les noves funcionalitats ha demostrat una gran eficàcia en el monitoratge i presa de decisions sobre equips. Quan es detecta la necessitat d'actuar sobre determinats equips, la plataforma envia les ordres corresponents al nou microservei desenvolupat en aquest projecte, l'ApiPulse, el qual executa aquestes ordres de manera eficient. Aquest enfocament ha permès una reducció significativa en els temps de resolució d'errors i ha millorat la disponibilitat dels equips instal·lats.

A més, s'ha integrat la possibilitat d'actuar sota demanda directament sobre els equips, oferint als tècnics una eina potent i flexible per a la gestió d'incidències. Aquesta funcionalitat s'ha integrat amb la plataforma AlphaMonitor, assegurant una experiència d'usuari coherent i eficient.

S'han desenvolupat les interfícies gràfiques adients, sempre treballant amb la visió del departament tècnic i la del departament de R+D+I alineades, millorant la usabilitat i l'eficiència dels tècnics en la seva tasca diària. Aquestes interfícies permeten una interacció intuïtiva amb el sistema, facilitant la cerca d'informació i la gestió d'errors.

En conclusió, el projecte ha assolit els objectius proposats, millorant la gestió d'equips remots i proporcionant eines avançades per als tècnics, tot plegat dins d'un sistema eficient i ben integrat amb la plataforma existent.

En termes de valoració personal, aquest projecte ha estat una experiència d'aprenentatge immensament enriquidora.

S'han adquirit habilitats des de la creació d'un microservei des de zero, la configuració i ús de Docker mitjançant la creació del Dockerfile d'ApiPulse, fins a la integració del microservei en un servei existent i la seva publicació a AWS. A més, s'han après tècniques de control de versions, imprescindibles per a una gestió eficient del desenvolupament de software.

Aquest procés no només ha servit per ampliar els coneixements tècnics, sinó que també ha reforçat la capacitat per afrontar i superar els reptes que es presenten en un projecte de desenvolupament de software complex, a agafar requeriments del client i aprendre a plantejar com ha de ser l'arquitectura i el software que ho fa possible, i a treballar conjuntament i en cohesió amb un altre departament dintre de la mateixa empresa.

10. Bibliografia

[1] Alphanet [en línia] [consulta: 20 de desembre de 2023]. Disponible a https://alphanet.cat/es/

[2] AlphaDataManager [en línia] [consulta: 21 de desembre de 2023]. Disponible a https://alphanet.cat/es/soluciones/alpha-data-manager/

[3] AlphaAlert, Aplicació d'Alphanet [en línia] [consulta: 21 de desembre de 2023]. Disponible a https://alphanet-solutions.com/es/alpha-alert/

[4] JIRA Software, Eina de seguiment de projectes [en línia] [consulta: 23 de desembre de 2023]. Disponible a https://www.atlassian.com/es/software/jira

[5] IntelliJ IDEA, Entorn de desenvolupament integrat per a Java i Kotlin [en línia] [consulta: 23 de desembre de 2023]. Disponible a https://www.jetbrains.com/help/idea/discover-intellij-idea.html

[6] Spring Boot, Eina facilitadora pel desenvolupament d'aplicacions web i microserveis amb Spring Framework [en línia] [consulta: 23 de desembre de 2023]. Disponible a https://spring.io/projects/spring-boot/

[7] Angular, Framework de JavaScript de tipus open source crear i programar aplicacions web [en línia] [consulta: 23 de desembre de 2023]. Disponible a https://angular.io

[8] PostgreSQL, Base de dades de codi obert de reputació sòlida per la seva fiabilitat, flexibilitat i suport d'estàndards tècnics oberts [en línia] [consulta: 23 de desembre de 2023]. Disponible a https://www.postgresql.org

[9] DBeaver, Software per a la gestió de bases de dades [en línia] [consulta: 23 de desembre de 2023]. Disponible a https://dbeaver.io

[10] Docker, Projecte de codi obert que automatitza el desplegament d'aplicacions dins de contenidors de programari [en línia] [consulta: 23 de desembre de 2023]. Disponible a https://www.docker.com

[11] GitLab, Gestor de repositoris Git que permet que els equips col·laborin en codis informàtics [en línia] [consulta: 23 de desembre de 2023]. Disponible a https://about.gitlab.com

[12] OpenSearch, Software flexible, escalable i de codi obert de crear solucions per a aplicacions intensives en dades [en línia] [consulta: 26 de desembre de 2023]. Disponible a https://opensearch.org

[13] AWS, Plataforma de serveis de núvol [en línia] [consulta: 28 de desembre de 2023]. Disponible a https://aws.amazon.com/es/

[14] Obsidian, Aplicació flexible per prendre notes i documentar projectes [en línia] [consulta: 1 de maig de 2024]. Disponible a https://obsidian.md/

[15] Markdown, Llenguatge de marcat lleuger [en línia] [consulta: 5 de maig de 2024]. Disponible a https://en.wikipedia.org/wiki/Markdown