

**ÁREA TECNOLOGÍAS DE INFORMACIÓN Y CIBERSEGURIDAD**

**IMPLEMENTACIÓN DE UN SISTEMA DE GESTIÓN DE MANTENIMIENTO COMPUTARIZADO PREDICTIVO CON ASISTENCIA DE INTELIGENCIA ARTIFICAL Y BOT OMNICANAL PARA SOMACOR S.A.**

Autores

**GALLARDO TRASLAVIÑA, LUCAS ADOLFO**

**MORALES ALCAVIL, MATÍAS JOSÉ**

Proyecto de título presentada al Instituto Profesional INACAP para optar al título profesional de Ingeniero en Informática

Profesor Guía

**Víctor Valenzuela Ruz**

Diciembre, 2025

Copiapó, Chile

©2025, Lucas Gallardo Traslaviña, Matías Morales Alcavil

Ninguna parte de esta tesis puede reproducirse o transmitirse bajo ninguna forma o por ningún medio o procedimiento, sin permiso por escrito de los autores.

Fecha Firma\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Fecha Firma\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Copiapó, 23 de diciembre de 2022

Yo Matias Morales Alcavil, Lucas Gallardo Traslaviña, RUT Nº 21.534.017-1,RUT LUCAS, domiciliado en Gino Bianchi Albasini 1066, Copiapó, DIRECCION LUCAS, Copiapó, estudiante de la sede Copiapó, del Programa de Estudio Ingeniería en Informática, autorizo al Instituto profesional INACAP a poner a disposición del público a texto completo la versión electrónica de mi Proyecto de Titulo titulada “IMPLEMENTACIÓN DE UN SISTEMA DE GESTIÓN DE MANTENIMIENTO COMPUTARIZADO PREDICTIVO CON ASISTENCIA DE INTELIGENCIA ARTIFICAL Y BOT OMNICANAL PARA SOMACOR S.A**”,** para fines académicos y servicios web de la Red de Bibliotecas INACAP u otros.

(Firma)

(Firma)

(Firma)



**ÁREA TECNOLOGÍAS DE INFORMACIÓN Y CIBERSEGURIDAD**

**IMPLEMENTACIÓN DE UN SISTEMA DE GESTIÓN DE MANTENIMIENTO COMPUTARIZADO PREDICTIVO CON ASISTENCIA DE INTELIGENCIA ARTIFICAL Y BOT OMNICANAL PARA SOMACOR S.A.**

Autores

**GALLARDO TRASLAVIÑA, LUCAS ADOLFO**

**MORALES ALCAVIL, MATÍAS JOSÉ**

|  |  |  |
| --- | --- | --- |
| Nombre | Calificación | Firma |
|  |  |  |
|  |  |  |

Profesor Guía

**Víctor Valenzuela Ruz**

Diciembre, 2025

Copiapó, Chille

Resumen

Esta tesis aborda el diseño e implementación de un Sistema de Gestión de Mantenimiento Computarizado (CMMS) con capacidades predictivas para la empresa SOMACOR S.A., prestadora de servicios en la faena minera La Coipa. El proyecto responde a la necesidad crítica de transformar un modelo de gestión reactivo, basado en registros manuales y con baja trazabilidad, hacia un esquema proactivo impulsado por datos.

La solución tecnológica integra una arquitectura de software desacoplada, utilizando un motor de Inteligencia Artificial basado en el algoritmo *Random Forest* para predecir fallas en la flota de equipos críticos. Dado el escenario de carencia inicial de datos históricos ("Cold Start"), se implementó una estrategia de generación de datos sintéticos basada en reglas expertas. Adicionalmente, el sistema incorpora un Bot Omnicanal en Telegram para agilizar la gestión de alertas y órdenes de trabajo en tiempo real. Los resultados proyectados apuntan a una optimización de la disponibilidad mecánica, reducción de costos por inactividad y mejora en la seguridad operacional.

Palabras clave: *Mantenimiento Predictivo, Inteligencia Artificial, Random Forest, CMMS, Omnicanalidad, Gestión de Activos, Minería, Cold Start, Trazabilidad Digital.*

Abstract

This thesis focuses on the design and implementation of a Computerized Maintenance Management System (CMMS) with predictive capabilities for SOMACOR S.A., a service provider at the La Coipa mining site. The project addresses the critical need to transform a reactive management model, based on manual records and low traceability, into a proactive, data-driven scheme.

The technological solution integrates a decoupled software architecture, using an Artificial Intelligence engine based on the Random Forest algorithm to predict failures in the critical equipment fleet. Given the initial lack of historical data ("Cold Start" scenario), a synthetic data generation strategy based on expert rules was implemented. Additionally, the system incorporates an Omnichannel Bot on Telegram to streamline alert management and work orders in real-time. The projected results point to an optimization of mechanical availability, reduction of downtime costs, and improvement in operational safety.

*Keywords: Predictive Maintenance, Artificial Intelligence, Random Forest, CMMS, Omnichannel, Asset Management, Mining, Cold Start, Digital Traceability*

**Tabla de Contenido**

Contenido

[1 INTODUCCION 1](#_Toc216464814)

[1.1 Planteamiento del problema 1](#_Toc216464815)

[1.2 Relevancia del trabajo 3](#_Toc216464816)

[1.3 Motivos para la elección del tema 4](#_Toc216464817)

[1.4 Objetivos del trabajo 5](#_Toc216464818)

[1.4.1 Objetivo General 5](#_Toc216464819)

[1.4.2 Objetivos Específicos 5](#_Toc216464820)

[1.5 Hipótesis. 6](#_Toc216464821)

[2 MARCO TEÓRICO 7](#_Toc216464822)

[2.1 Gestión de Mantenimiento Industrial 7](#_Toc216464823)

[2.1.1 Evolución de las Estrategias de Mantenimiento 8](#_Toc216464824)

[2.1.2 Mantenimiento 4.0 y Digitalización 9](#_Toc216464825)

[2.1.3 Desafíos y Oportunidades en el Contexto Minero 10](#_Toc216464826)

[2.2 Inteligencia Artificial y Machine Learning 10](#_Toc216464827)

[2.2.1 Aprendizaje Supervisado vs. No Supervisado 11](#_Toc216464828)

[2.2.2 Algoritmo Random Forest 12](#_Toc216464829)

[2.2.3 Problemática del "Cold Start" (Arranque en Frío) 13](#_Toc216464830)

[2.2.4 Generación de Datos Sintéticos basados en Reglas Expertas 13](#_Toc216464831)

[2.3 Arquitectura de Software Moderna 14](#_Toc216464832)

[2.3.1 Arquitectura Monolítica vs. Desacoplada (Headless) 14](#_Toc216464833)

[2.3.2 APIs RESTful y Protocolos de Comunicación 15](#_Toc216464834)

[2.3.3 Tecnologías Web Seleccionadas 15](#_Toc216464835)

[2.4 Omnicanalidad y Experiencia de Usuario (UX) 16](#_Toc216464836)

[2.4.1 Concepto de Omnicanalidad 17](#_Toc216464837)

[2.4.2 Interfaces Conversacionales (Chatbots) en la Industria 17](#_Toc216464838)

[2.4.3 Reducción de la Fricción Tecnológica 18](#_Toc216464839)

[3 FORMULACIÓN Y DELIMITACIÓN DEL PROBLEMA EN ESTUDIO 19](#_Toc216464840)

[3.1 Descripción de la Organización 19](#_Toc216464841)

[3.1.1 Antecedentes de la Empresa 19](#_Toc216464842)

[3.1.2 Misión y Visión 20](#_Toc216464843)

[3.1.3 Estructura Organizacional 20](#_Toc216464844)

[3.1.4 Localización y Entorno Operativo 21](#_Toc216464845)

[3.1.5 Infraestructura Tecnológica y Conectividad en Faena 22](#_Toc216464846)

[3.2 Análisis de la Situación Actual (As-Is) 23](#_Toc216464847)

[3.2.1 Descripción del Proceso de Mantenimiento Actual 23](#_Toc216464848)

[3.2.2 Diagrama de Flujo del Proceso Actual 25](#_Toc216464849)

[3.2.3 Identificación de Deficiencias Críticas 25](#_Toc216464850)

[3.3 Necesidades del Negocio y Requerimientos del Sistema 26](#_Toc216464851)

[3.3.1 tabla de Requerimientos Funcionales Clave 26](#_Toc216464852)

[3.3.2 tabla de Requerimientos Funcionales Clave 28](#_Toc216464853)

[4 METODOLOGÍA DE TRABAJO Y PLANIFICACIÓN 30](#_Toc216464854)

[4.1 Selección de la Metodología Ágil (Scrum) 30](#_Toc216464855)

[4.1.1 Justificación de la Elección 31](#_Toc216464856)

[4.2 Definición de Roles y Responsabilidades 32](#_Toc216464857)

[4.2.1 Product Owner (Dueño del Producto) 32](#_Toc216464858)

[4.2.2 Scrum Master (Facilitador) 32](#_Toc216464859)

[4.2.3 Equipo de Desarrollo (Development Team) 32](#_Toc216464860)

[4.3 Planificación y Cronograma de Trabajo 34](#_Toc216464861)

[4.3.1 Fase 1: Análisis y Definición de Requerimientos (Sprint 1) 34](#_Toc216464862)

[4.3.2 Fase 2: Diseño de Arquitectura y Datos (Sprint 2) 35](#_Toc216464863)

[4.3.3 Fase 3: Construcción y Desarrollo (Sprints 3-4) 35](#_Toc216464864)

[4.3.4 Fase 4: Pruebas y Aseguramiento de Calidad (Sprint 5) 36](#_Toc216464865)

[4.3.5 Fase 5: Implementación y Cierre (Sprint 6) 36](#_Toc216464866)

[5 EVALUACIÓN DE FACTIBILIDAD 40](#_Toc216464867)

[5.1 Factibilidad Técnica 40](#_Toc216464868)

[5.1.1 Tecnologías de Desarrollo y Librerías (*Stack*) 41](#_Toc216464869)

[5.1.2 Requisitos de Infraestructura y Despliegue (*Cloud*) 41](#_Toc216464870)

[5.1.3 Requisitos del Usuario Final 42](#_Toc216464871)

[5.2 Factibilidad Económica 43](#_Toc216464872)

[5.2.1 Estructura de Inversión Inicial (CAPEX) 43](#_Toc216464873)

[5.2.2 Presupuesto Operacional (Ingresos y Egresos) 44](#_Toc216464874)

[5.2.3 Flujo de Caja Proyectado 45](#_Toc216464875)

[5.2.4 Indicadores de Rentabilidad y Análisis Financiero 46](#_Toc216464876)

[5.2.5 Justificación del Modelo de Negocio Propuesto 47](#_Toc216464877)

[5.3 Factibilidad Legal 48](#_Toc216464878)

[5.3.1 Normativa Laboral y Seguridad 48](#_Toc216464879)

[5.3.2 Propiedad Intelectual y Licenciamiento 48](#_Toc216464880)

[5.4 Factibilidad Operacional 49](#_Toc216464881)

[5.4.1 Gestión del Cambio y Adopción 49](#_Toc216464882)

[5.4.2 Disponibilidad del Recurso Humano 49](#_Toc216464883)

[5.5 Gestión de Riesgos y Plan de Contingencia 50](#_Toc216464884)

[5.5.1 Identificación de Riesgos 50](#_Toc216464885)

[5.5.2 Matriz de riesgo 51](#_Toc216464886)

[5.5.3 Análisis Cualitativo 54](#_Toc216464887)

[5.5.4 Análisis Cuantitativo 59](#_Toc216464888)

[5.5.5 Planes de respuesta 64](#_Toc216464889)

[6 INGENIERÍA DEL SISTEMA 69](#_Toc216464890)

[6.1 Especificación de Requisitos del Sistema 69](#_Toc216464891)

[6.1.1 Requisitos Funcionales (RF) 70](#_Toc216464892)

[6.1.2 Requisitos No Funcionales (RNF) 73](#_Toc216464893)

[6.2 Modelado de Datos y Diagramas de Ingeniería 75](#_Toc216464894)

[6.2.1 Modelo de Datos Relacional (DER) 75](#_Toc216464895)

[6.2.2 Diccionario de Datos 79](#_Toc216464896)

[6.2.3 Modelado de Procesos de Negocio (BPMN To-Be) 85](#_Toc216464897)

[6.2.4 Diagrama de Clases y Lógica de Negocio 87](#_Toc216464898)

[6.3 Diseño de la Arquitectura del *Software* 88](#_Toc216464899)

[6.3.1 Diagrama de Infraestructura y Despliegue 89](#_Toc216464900)

[6.3.2 Descripción de Componentes 90](#_Toc216464901)

[6.3.3 Flujo de Comunicación Omnicanal 91](#_Toc216464902)

[6.4 Diseño de Interfaz de Usuario (UI/UX) 91](#_Toc216464903)

[6.4.1 Tablero de Control Operacional (*Dashboard*) 92](#_Toc216464904)

[6.4.2 Gestión de Activos y Hoja de Vida Digital 94](#_Toc216464905)

[6.4.3 Flujo de Órdenes de Trabajo (OT) 95](#_Toc216464906)

[6.4.4 Interfaz Omnicanal (Telegram Bot) 96](#_Toc216464907)

[7 CONCLUSIONES Y RECOMENDACIONES 98](#_Toc216464908)

[7.1 Conclusiones 98](#_Toc216464909)

[7.2 Recomendaciones para la Organización 100](#_Toc216464910)

[7.3 Lecciones Aprendidas 101](#_Toc216464911)

[7.4 Trabajos Futuros 102](#_Toc216464912)

[8 Bibliografía 104](#_Toc216464913)

**Índices de Figuras**

[Figura 1 Organigrama Corporativo de SOMACOR S. A Fuente: Elaboración propia 21](#_Toc216464914)

[Figura 2 Ubicación geográfica de la faena La Coipa 22](#_Toc216464915)

[Figura 3 Diagrama flujo de proceso actual 25](#_Toc216464916)

[Figura 4 Carta Gantt del Proyecto 36](#_Toc216464917)

[Figura 5 Carta Gantt del proyecto Fase 1 37](#_Toc216464918)

[Figura 6 Carta Gantt del proyecto Fase 2 37](#_Toc216464919)

[Figura 7 Carta Gantt del proyecto Fase 3 38](#_Toc216464920)

[Figura 8 Carta Gantt del proyecto Fase 4 38](#_Toc216464921)

[Figura 9 Carta Gantt del proyecto Fase 5 39](#_Toc216464922)

[Figura 10 Diagrama Entidad-Relación – DER 76](#_Toc216464923)

[Figura 11 Diagrama BPMN del Ciclo de Mantenimiento Predictivo Propuesto (To-Be). 86](#_Toc216464924)

[Figura 12 Diagrama de Clases UML 88](#_Toc216464925)

[Figura 13 Diagrama de Arquitectura de Despliegue 89](#_Toc216464926)

[Figura 14 Captura de pantalla del Dashboard Principal 92](#_Toc216464927)

[Figura 15 Captura de pantalla del Dashboard Principal parte 2 93](#_Toc216464928)

[Figura 16 Captura de pantalla del Dashboard Principal parte 3 93](#_Toc216464929)

[Figura 17 Captura de pantalla del Perfil de un Activo/Equipo 94](#_Toc216464930)

[Figura 18 Captura de pantalla de la Gestión de OTs 95](#_Toc216464931)

[Figura 19 Captura de pantalla de la interacción con el Bot en Telegram parte 2 96](#_Toc216464932)

[Figura 20 Captura de pantalla de la interacción con el Bot en Telegram 96](#_Toc216464933)

[Figura 21 Captura de pantalla de la interacción con el Bot en Telegram parte 3 97](#_Toc216464934)

**Índices de Tablas**

[Tabla 1 Requerimientos Funcionales Clave 26](#_Toc216464935)

[Tabla 2 Requerimientos No Funcionales Clave 28](#_Toc216464936)

[Tabla 3 Roles y Responsabilidades del Proyecto 33](#_Toc216464937)

[Tabla 4 Requerimientos de Infraestructura en la Nube 41](#_Toc216464938)

[Tabla 5 Inversión Inicial (Puesta en Marcha) 43](#_Toc216464939)

[Tabla 6 Estructura Mensual de Ingresos y Egresos 44](#_Toc216464940)

[Tabla 7 Resumen del Flujo de Caja (en CLP) 45](#_Toc216464941)

[Tabla 8 Indicadores de Viabilidad Económica 46](#_Toc216464942)

[Tabla 9 Criterios de evaluación 51](#_Toc216464943)

[Tabla 10 Niveles de los criterios de evaluación 52](#_Toc216464944)

[Tabla 11 Matriz de riesgos 53](#_Toc216464945)

[Tabla 12 Criterios del análisis cualitativo 54](#_Toc216464946)

[Tabla 13 Análisis cualitativo de la matriz de riesgos 56](#_Toc216464947)

[Tabla 14 Análisis Comparativo Cualitativo 58](#_Toc216464948)

[Tabla 15 Criterios del análisis cuantitativo 59](#_Toc216464949)

[Tabla 16 Análisis cuantitativo de la matriz de riesgos 61](#_Toc216464950)

[Tabla 17 Análisis Comparativo Cualitativo 64](#_Toc216464951)

[Tabla 18 Detalle de Requerimientos Funcionales 70](#_Toc216464952)

[Tabla 19 Detalle de Requerimientos No Funcionales 73](#_Toc216464953)

[Tabla 20 Ubicación 79](#_Toc216464954)

[Tabla 21 Activo 79](#_Toc216464955)

[Tabla 22 Plan\_Mantenimiento 80](#_Toc216464956)

[Tabla 23 Orden\_Trabajo 81](#_Toc216464957)

[Tabla 24 Plantilla\_Lista 81](#_Toc216464958)

[Tabla 25 Lista\_Verificacion\_Repuestos 82](#_Toc216464959)

[Tabla 26 Pieza\_Repuesto 82](#_Toc216464960)

[Tabla 27 Movimiento\_Stock 83](#_Toc216464961)

[Tabla 28 Prediccion\_Fallos 83](#_Toc216464962)

[Tabla 29 Alerta 84](#_Toc216464963)

[Tabla 30 Usuario 84](#_Toc216464964)

[Tabla 31 Rol 85](#_Toc216464965)

[Tabla 32 Notificacion 85](#_Toc216464966)

**INTODUCCION**

## Planteamiento del problema

La industria minera en Chile constituye uno de los pilares fundamentales de la economía nacional, operando bajo estándares de exigencia extremadamente rigurosos donde la continuidad operacional no es solo un objetivo, sino un imperativo estratégico. En este contexto, las empresas proveedoras de servicios juegan un rol crucial en la cadena de valor. Específicamente, la empresa **SOMACOR S.A.**, con más de cinco décadas de trayectoria, se desempeña como un socio estratégico en la faena minera **La Coipa**, operada por la transnacional Gold Fields.

Esta operación se sitúa geográficamente sobre los 3.000 metros sobre el nivel del mar, un entorno hostil caracterizado por la oscilación térmica, la presencia constante de polvo en suspensión y terrenos con pendientes abruptas. Estas variables ambientales someten a la flota de maquinaria pesada y vehículos de apoyo —tales como cargadores frontales, equipos *Supersucker* y camionetas de supervisión— a un desgaste mecánico acelerado, incrementando exponencialmente la probabilidad de fallas funcionales.

Sin embargo, el diagnóstico de la situación actual revela una brecha técnica crítica respecto a los estándares de la industria. Mientras que el promedio de **disponibilidad mecánica** aceptado en la gran minería para equipos de apoyo oscila entre el **85% y el 90%**, las proyecciones internas sugieren que Somacor opera actualmente con una disponibilidad inferior a este estándar debido a su gestión reactiva. Esta discrepancia porcentual no es solo un indicador técnico, sino que representa una vulnerabilidad estratégica que pone en riesgo la competitividad del contrato frente a las exigencias de Gold Fields.

Actualmente, los procesos de inspección y control de la flota se caracterizan por ser predominantemente manuales. La operación depende de registros físicos, específicamente *checklists* en papel, lo que obliga a cada operador a invertir tiempo valioso —estimado en 10 minutos diarios por equipo— en el llenado manual de formularios. Esta metodología analógica genera lo que se denomina técnicamente como una "ceguera de datos" o caja negra operativa: la información crítica sobre el estado de los equipos queda atrapada en archivadores físicos, descentralizada y desconectada de los tomadores de decisiones.

Al no existir un monitoreo en tiempo real ni un análisis de tendencias de falla, la organización opera bajo un esquema correctivo de emergencia, interviniendo los equipos solo cuando la falla ya se ha manifestado. Las consecuencias financieras de este modelo son devastadoras. Datos internos de la operación indican que la detención no programada (*downtime*) de un activo crítico, como un equipo *Supersucker*, conlleva un costo operacional por lucro cesante de aproximadamente **$517.000 pesos diarios**. A esto se suman las multas contractuales aplicadas por el cliente mandante, las cuales pueden ascender a 5 UF diarias, totalizando pérdidas que superan los **$710.500 por día de inactividad** (SOMACOR S.A., 2024).

Además del impacto económico, la desconexión entre la detección del síntoma en terreno y la acción correctiva administrativa genera cuellos de botella logísticos. En escenarios de fallas mayores, la falta de previsión en el stock de repuestos puede extender los tiempos de reparación entre una y dos semanas, elevando las pérdidas acumuladas por un solo evento sobre los 10 millones de pesos. Este escenario pone en riesgo no solo la rentabilidad del contrato, sino también el cumplimiento de los Acuerdos de Nivel de Servicio (SLA) comprometidos y la seguridad de los operadores en zonas de alto riesgo geográfico.

## Relevancia del trabajo

La transición hacia la denominada Industria 4.0 (Schwab, 2016) exige que las empresas colaboradoras del sector minero evolucionen desde ser meros proveedores de servicios hacia convertirse en gestores de inteligencia operacional. La relevancia de este proyecto radica en su capacidad para transformar estructuralmente el modelo de gestión de activos de SOMACOR S.A., cerrando la brecha digital detectada y posicionando a la empresa a la vanguardia tecnológica mediante una solución integral.

El trabajo propuesto es relevante por las siguientes razones estratégicas:

**Impacto Económico y Operacional:** La implementación de un CMMS con capacidades predictivas permite un cambio de paradigma fundamental: transitar desde la filosofía reactiva de "reparar tras la falla" hacia una proactiva de "predecir y prevenir". Al instrumentalizar los datos operativos, el sistema anticipa condiciones de riesgo mecánico antes de que se conviertan en fallas catastróficas. Esto incide directamente en la reducción del *downtime*, la optimización de la vida útil de los activos y la disminución de los costos asociados a reparaciones de emergencia y multas contractuales, protegiendo así la rentabilidad del contrato.

**Seguridad y Salud Ocupacional (ISO 45001):** El proyecto posee una relevancia humana crítica. Al asegurar que la maquinaria opere dentro de parámetros técnicos óptimos y predecir posibles colapsos mecánicos, se reduce significativamente la exposición de los trabajadores a accidentes laborales derivados de fallas imprevistas en entornos geográficos hostiles de alta montaña. Esto alinea la operación tecnológica con los estándares del Sistema de Gestión Integrado (SGI) de la organización (International Organization for Standardization, 2018), priorizando la integridad física de los operadores.

**Innovación Tecnológica y Técnica:** Este trabajo aborda un desafío técnico complejo y común en la industria: el problema del "Arranque en Frío" (*Cold Start*) (Schein & Popescul, Methods and metrics for cold-start recommendations, 2002). La metodología propuesta para generar datos sintéticos basados en conocimiento experto demuestra cómo organizaciones con baja madurez digital pueden adoptar tecnologías de vanguardia, como Inteligencia Artificial y *Machine Learning*, sin necesidad de esperar años para construir un histórico de datos. Esto establece un modelo tecnológico escalable y replicable para otras empresas de servicios industriales que enfrentan barreras similares.

## Motivos para la elección del tema

La elección de este tema de investigación surge de la convergencia entre una necesidad industrial latente y la observación directa de una realidad operativa que requiera modernización inmediata. Si bien la ingeniería informática ofrece herramientas de vanguardia, su aplicación en entornos hostiles a menudo se ve relegada por procesos tradicionales.

El detonante principal para este proyecto fue la observación en terreno de la fricción existente en los procesos actuales. Fue evidente cómo operadores expertos, trabajando sobre los 3.000 m.s.n.m., debían interrumpir labores críticas y quitarse los guantes de seguridad —exponiéndose al frío extremo y al polvo en suspensión— para diligenciar planillas de papel que terminaban sucias o ilegibles. Esta burocracia analógica no solo entorpecía la excelencia técnica, sino que desincentivaba el reporte oportuno de fallas, creando la "ceguera de datos" que afecta a la organización.

Motiva el desarrollo de este trabajo el desafío técnico de diseñar una solución que elimine esta fricción. No se trata solo de digitalizar un formulario, sino de integrar disciplinas complejas —arquitectura de software desacoplada (*Headless*), ciencia de datos (*Machine Learning*) y omnicanalidad— para crear una herramienta que se adapte al entorno del trabajador. El objetivo es demostrar que la tecnología avanzada puede ser accesible, transformando el reporte de mantenimiento en una acción tan sencilla como enviar un mensaje por Telegram, empoderando así al personal en terreno y generando valor tangible para SOMACOR S.A.

## Objetivos del trabajo

### Objetivo General

Diseñar e implementar una plataforma integral de gestión de mantenimiento (CMMS) para la flota de equipos críticos en la faena La Coipa, que integre capacidades de predicción de fallos mediante algoritmos de Aprendizaje Automático y comunicación omnicanal, con el fin de incrementar la disponibilidad mecánica, asegurar la continuidad operativa y optimizar los costos operativos asociados a detenciones no programadas.

### Objetivos Específicos

Para alcanzar el objetivo general, se han establecido las siguientes metas específicas:

* **levantamiento de información:** Desarrollar e implementar interfaces web y móviles que permitan la captura digital de *checklists* y bitácoras de operación, eliminando el uso de papel y garantizando la trazabilidad inmediata de la condición de los activos en una base de datos centralizada.
* **Análisis y diseño:** Análisis y diseño un modelo de Inteligencia Artificial capaz de operar en escenarios de escasez de datos (*Cold Start*), utilizando generación de datos sintéticos y el algoritmo *Random Forest* para la detección temprana de anomalías.
* **Programación de la solución:** Desarrollar aplicación web e Integrar un Bot Omnicanal en la plataforma Telegram que automatice la distribución de alertas críticas y la gestión de Órdenes de Trabajo (OT) en tiempo real, reduciendo la latencia comunicacional entre los supervisores y el personal técnico en terreno.
* **Implementación:** Implementar un cuadro de mando (*dashboard*) gerencial que visualice indicadores clave de desempeño (KPIs) como la disponibilidad física y la confiabilidad, permitiendo a la gerencia gestionar la flota basándose en datos objetivos y evidencia cuantitativa.

## Hipótesis.

La implementación de un Sistema de Gestión de Mantenimiento Computarizado (CMMS) que integre modelos de predicción de fallos basados en Inteligencia Artificial y un sistema de notificación omnicanal, permitirá reducir los tiempos de inactividad no programada (*downtime*) de la flota crítica de SOMACOR S.A. en la faena La Coipa, al mejorar la detección temprana de anomalías y la velocidad de respuesta operativa en comparación con el actual sistema de gestión manual reactivo.

**MARCO TEÓRICO**

La gestión de activos físicos en la minería moderna enfrenta el desafío crítico de maximizar la disponibilidad operacional en entornos de creciente complejidad técnica y geográfica. Para abordar esta problemática de manera efectiva, resulta imperativo fundamentar la solución tecnológica propuesta sobre bases teóricas sólidas que integren los principios de la ingeniería de mantenimiento con las capacidades avanzadas de las ciencias de la computación.

El presente capítulo establece los fundamentos conceptuales y técnicos que sustentan el diseño y desarrollo de la plataforma CMMS predictiva. En primera instancia, se analizará la evolución de los paradigmas de mantenimiento industrial, contextualizando la necesaria transición desde enfoques reactivos hacia estrategias predictivas enmarcadas en la Industria 4.0.

Posteriormente, se profundizará en los principios de la Inteligencia Artificial y el Aprendizaje Automático (*Machine Learning*), justificando la selección de algoritmos de aprendizaje supervisado y las metodologías de generación de datos sintéticos para mitigar la carencia de registros históricos (*Cold Start*). Finalmente, se describirán los estándares de arquitectura de software moderna y los conceptos de omnicanalidad que habilitan la operatividad y usabilidad del sistema en condiciones de faena.

## Gestión de Mantenimiento Industrial

El mantenimiento industrial ha dejado de ser considerado una actividad de soporte periférica para convertirse en un pilar estratégico de la competitividad empresarial. En sectores intensivos en activos físicos, como la minería, la gestión eficiente del ciclo de vida de la maquinaria es determinante para la sostenibilidad financiera y operativa del negocio. Esta sección analiza la evolución de las filosofías de mantenimiento y su convergencia con las tecnologías digitales en el paradigma de la Industria 4.0.

### Evolución de las Estrategias de Mantenimiento

La gestión de activos ha transitado por distintas generaciones o estadios de madurez, cada uno definido por la disponibilidad tecnológica y la comprensión de los modos de falla de los equipos.

* **Mantenimiento Correctivo (Reactivo):** También conocido como "operar hasta la falla" (*run-to-failure*). Esta estrategia se basa en intervenir el activo únicamente cuando ha perdido su función operativa. Si bien minimiza los costos de planificación inicial, conlleva altos niveles de incertidumbre y costos indirectos severos asociados al lucro cesante por detenciones no programadas (*downtime*). Históricamente, fue el estándar hasta mediados del siglo XX, pero resulta insostenible en operaciones modernas de flujo continuo.
* **Mantenimiento Preventivo (Basado en Tiempo/Uso):** Surge como respuesta a la necesidad de mayor confiabilidad. Este modelo programa intervenciones en intervalos fijos (horas de operación, kilómetros o tiempo calendario), bajo el supuesto estadístico de que la probabilidad de falla aumenta linealmente con la edad del componente (curva de la "bañera"). Aunque reduce la ocurrencia de fallas catastróficas, introduce ineficiencias como el "sobre-mantenimiento" (reemplazo de piezas aún útiles) y no aborda las fallas aleatorias que no dependen del tiempo de uso.
* **Mantenimiento Predictivo (Basado en Condición):** Representa el estado del arte en la gestión de activos. Se fundamenta en el monitoreo constante de la condición real del equipo (vibraciones, temperatura, análisis de fluidos) para pronosticar fallas futuras antes de que se manifiesten funcionalmente (Mobley, 2002). Esta estrategia permite maximizar la vida útil de los componentes y programar intervenciones "justo a tiempo", optimizando la disponibilidad mecánica y reduciendo drásticamente los costos de inventario y reparación.

### Mantenimiento 4.0 y Digitalización

La llegada de la **Cuarta Revolución Industrial (Industria 4.0)** ha redefinido el alcance del mantenimiento predictivo mediante la integración de tecnologías habilitadoras como el Internet de las Cosas (IoT), la Computación en la Nube (*Cloud Computing*) y la Inteligencia Artificial.

El concepto de **Mantenimiento 4.0** trasciende el monitoreo de condición tradicional al incorporar la capacidad de procesar grandes volúmenes de datos (*Big Data*) provenientes de múltiples fuentes. Esto permite transitar desde un análisis descriptivo ("¿qué pasó?") hacia uno prescriptivo ("¿qué debemos hacer para que no pase?"). En este nuevo paradigma, los Sistemas de Gestión de Mantenimiento Computarizado (CMMS) dejan de ser simples repositorios de registros para convertirse en plataformas inteligentes capaces de orquestar flujos de trabajo automatizados y apoyar la toma de decisiones en tiempo real.

### Desafíos y Oportunidades en el Contexto Minero

La industria minera presenta desafíos operacionales únicos que condicionan la implementación de estrategias de mantenimiento avanzado. Factores como la ubicación remota de los yacimientos, la geografía abrupta y las condiciones ambientales extremas (oscilación térmica, polvo en suspensión, hipoxia) imponen un desgaste acelerado a la maquinaria, haciendo que las curvas de degradación estándar a menudo no sean aplicables.

En este contexto, la adopción de tecnologías digitales enfrenta barreras específicas, tales como la intermitencia en la conectividad de redes y la dificultad para desplegar hardware delicado en terreno. Sin embargo, el potencial de optimización es significativo: la capacidad de predecir fallas en activos críticos no solo impacta en la rentabilidad del contrato mediante la reducción de multas y costos operativos, sino que es fundamental para la seguridad laboral, al disminuir la exposición de los mantenedores a intervenciones de emergencia en zonas de alto riesgo.

## Inteligencia Artificial y Machine Learning

La Inteligencia Artificial (IA) se define como la simulación de procesos de inteligencia humana por parte de sistemas informáticos, abarcando el aprendizaje, el razonamiento y la autocorrección (Russell & Norvig, 2016). Dentro de este vasto campo, el Aprendizaje Automático (*Machine Learning*) emerge como una subdisciplina que dota a los sistemas de la capacidad de aprender y mejorar automáticamente a partir de la experiencia sin ser explícitamente programados para cada escenario.

Para el desarrollo del sistema CMMS predictivo, es fundamental comprender las tipologías de aprendizaje y los algoritmos específicos que permiten transformar datos operativos brutos en predicciones de falla accionables.

### Aprendizaje Supervisado vs. No Supervisado

En el contexto del modelado predictivo industrial, los algoritmos se clasifican principalmente en dos categorías según la naturaleza de los datos de entrenamiento:

* **Aprendizaje Supervisado:** En este paradigma, el algoritmo se entrena utilizando un conjunto de datos etiquetado, donde cada entrada (*input*) está asociada a una salida correcta conocida (*target*). El objetivo es aprender una función de mapeo que pueda predecir la etiqueta de salida para nuevos datos no vistos. Es el enfoque estándar para tareas de clasificación (ej. predecir si un equipo fallará o no) y regresión (ej. estimar la vida útil remanente).
* **Aprendizaje No Supervisado:** Se utiliza cuando los datos de entrenamiento no poseen etiquetas históricas. El sistema intenta identificar patrones, estructuras o anomalías ocultas en los datos por sí mismo. Si bien es útil para la segmentación exploratoria, carece de la direccionalidad específica requerida para predecir fallas críticas con la precisión que exige la operación minera.

Para este proyecto, se ha optado por un enfoque de **Aprendizaje Supervisado**, dado que el objetivo es clasificar el estado futuro de la maquinaria en categorías definidas de "Falla" o "No Falla" basándose en variables operativas conocidas.

### Algoritmo Random Forest

El núcleo predictivo de la plataforma se fundamenta en el algoritmo **Random Forest** (Bosque Aleatorio). Este es un método de aprendizaje por ensamble (*ensemble learning*) que opera construyendo una multitud de árboles de decisión durante la fase de entrenamiento y generando como salida la clase que es el modo de las clases (clasificación) o la predicción media (regresión) de los árboles individuales (Breiman, 2001).

La selección de *Random Forest* frente a otras alternativas avanzadas, como las Redes Neuronales Profundas (*Deep Learning*), se justifica por tres factores técnicos decisivos para el entorno de SOMACOR S.A.:

1. **Robustez ante "Small Data":** A diferencia del *Deep Learning*, que requiere volúmenes masivos de datos para converger y evitar el sobreajuste, *Random Forest* ofrece un rendimiento generalizable superior con conjuntos de datos limitados o medianos, lo cual es característico en la fase inicial de digitalización de una faena.
2. **Manejo de No-Linealidad:** La maquinaria pesada opera bajo condiciones termodinámicas y mecánicas complejas. Este algoritmo es capaz de capturar relaciones no lineales entre variables (por ejemplo, la correlación no proporcional entre horas de uso y temperatura de fluidos) sin requerir una transformación exhaustiva de las características.
3. **Interpretabilidad (*Feature Importance*):** Una ventaja crítica es la capacidad del algoritmo para determinar la "importancia de las características". Esto permite al sistema no solo alertar sobre una falla, sino explicar *por qué* se generó la alerta (ej. "debido a un aumento anormal en la vibración"), facilitando la validación técnica y la confianza del personal humano en la herramienta.

### Problemática del "Cold Start" (Arranque en Frío)

Uno de los obstáculos más significativos en la implementación de IA en industrias tradicionales es el problema del ***Cold Start*** o arranque en frío. Este fenómeno ocurre cuando un sistema predictivo debe comenzar a operar sin disponer de un historial de datos suficiente para entrenar sus modelos (Schein, Popescul, Ungar, & Pennock, 2002). En el caso de la faena La Coipa, la gestión previa basada en registros manuales en papel generó una "ceguera de datos", resultando en la inexistencia de un repositorio digital estructurado con etiquetas históricas de fallas. Sin estos datos previos, un modelo supervisado convencional no puede aprender a distinguir patrones de riesgo, lo que tradicionalmente retrasaría la implementación del proyecto meses o años hasta recolectar suficiente data empírica.

### Generación de Datos Sintéticos basados en Reglas Expertas

Para superar la barrera del *Cold Start* y habilitar la operatividad del sistema desde el día uno, se implementó una estrategia de ingeniería de datos basada en la **Generación de Datos Sintéticos**.

Este enfoque consiste en la codificación programática del conocimiento heurístico de los ingenieros de confiabilidad y mecánicos expertos. En lugar de esperar a que ocurran fallas reales, se desarrollan algoritmos generativos que simulan el comportamiento de la maquinaria y crean un *dataset* inicial artificial. Estos datos sintéticos incorporan patrones de degradación conocidos y modos de fallo típicos, permitiendo entrenar al modelo *Random Forest* con escenarios teóricos validados.

Esta metodología permite desplegar un modelo funcional que actúa inicialmente como un "Sistema Experto Probabilístico", capaz de detectar condiciones de riesgo lógicas. Posteriormente, a medida que la operación real genere datos empíricos, el sistema iniciará un proceso de reentrenamiento evolutivo, reemplazando progresivamente los datos sintéticos por datos reales, aumentando así la precisión del modelo de manera orgánica.

## Arquitectura de Software Moderna

La efectividad de un modelo predictivo se ve severamente limitada si la información generada no llega de manera oportuna a los tomadores de decisiones. Por esta razón, el estado del arte en el desarrollo de sistemas de gestión industrial ha evolucionado desde estructuras monolíticas rígidas hacia arquitecturas distribuidas que priorizan la agilidad, la escalabilidad y la interoperabilidad entre múltiples plataformas.

### Arquitectura Monolítica vs. Desacoplada (Headless)

Tradicionalmente, las aplicaciones web se construían bajo una **Arquitectura Monolítica**, donde la lógica de negocio, el acceso a datos y la interfaz de usuario estaban entrelazados en una única base de código. Si bien este enfoque facilita el desarrollo inicial, presenta serias limitaciones de escalabilidad y flexibilidad, ya que cualquier cambio en la interfaz requiere la recompilación de todo el sistema.

Para garantizar la flexibilidad requerida por una operación minera dinámica, este proyecto adopta una **Arquitectura de Software Desacoplada** o *Headless Architecture*. Este paradigma de diseño separa estrictamente la capa de lógica de negocio y procesamiento de datos (*Backend*) de la capa de presentación (*Frontend*) (Fielding, 2000).

La principal ventaja teórica de este enfoque es la **omnicanalidad nativa**: un único *backend* puede servir datos simultáneamente a múltiples "cabezas" o interfaces, como un panel web administrativo, una aplicación móvil o, en este caso, un bot de mensajería, asegurando que la lógica de negocio y los datos sean consistentes en todos los puntos de contacto.

### APIs RESTful y Protocolos de Comunicación

La comunicación entre los componentes desacoplados se realiza a través de una **API RESTful** (*Representational State Transfer*). Este estilo arquitectónico define un conjunto de restricciones y protocolos para la creación de servicios web, permitiendo que sistemas heterogéneos se comuniquen a través del protocolo HTTP estándar.

En el contexto del CMMS, la API actúa como el puente universal. Expone los recursos del sistema (activos, órdenes de trabajo, predicciones) en formato JSON, permitiendo que tanto la interfaz web (React) como el Bot (Telegram) consuman y envíen información de manera estandarizada, segura y eficiente, independientemente del lenguaje de programación o dispositivo que utilice el cliente final.

### Tecnologías Web Seleccionadas

Para materializar esta arquitectura, se seleccionaron tecnologías que representan el estándar actual en el desarrollo de software de alto rendimiento:

* **Backend (Django - Python):** El núcleo lógico del sistema se sustenta en **Django**, un *framework* web de alto nivel escrito en Python (Django Software Foundation, 2023). Su elección se fundamenta teóricamente en su robustez ("baterías incluidas"), su seguridad intrínseca contra vulnerabilidades comunes (como inyección SQL) y, crucialmente, su integración nativa con el ecosistema de Inteligencia Artificial de Python (Scikit-learn, Pandas), lo que facilita la orquestación de los modelos predictivos *Random Forest* dentro del mismo flujo de servidor.
* **Frontend (React JS):** La interfaz de usuario administrativa se construye utilizando **React**, una biblioteca de JavaScript desarrollada por Facebook para construir interfaces de usuario interactivas (Meta Platforms, 2023). React permite la creación de **Aplicaciones de Página Única (SPA)**, donde el contenido se carga dinámicamente sin necesidad de recargar la página completa. Esto ofrece una experiencia de usuario (*User Experience - UX*) fluida y reactiva, similar a una aplicación de escritorio, minimizando la latencia percibida y el consumo de ancho de banda, factor crítico en la conectividad limitada de la faena.

## Omnicanalidad y Experiencia de Usuario (UX)

La implementación exitosa de un sistema de información en un entorno industrial no depende únicamente de la robustez de sus algoritmos, sino de la capacidad de los usuarios finales para interactuar con la tecnología de manera eficiente y sin resistencia. En este contexto, los conceptos de Omnicanalidad y Experiencia de Usuario (*User Experience - UX*) cobran una relevancia crítica para asegurar la adopción tecnológica en faena.

### Concepto de Omnicanalidad

La omnicanalidad es una estrategia de gestión de la comunicación que busca unificar la experiencia del usuario a través de diversos puntos de contacto, eliminando las barreras entre los canales físicos y digitales. A diferencia de la multicanalidad, donde los canales operan como silos independientes, la omnicanalidad garantiza que la información y el contexto se mantengan consistentes independientemente del medio utilizado para acceder al sistema.

En el diseño de sistemas modernos, esto implica que un evento crítico (como una predicción de falla) debe estar disponible simultáneamente para el gerente en su *dashboard* web y para el mecánico en su dispositivo móvil, permitiendo una transición fluida entre la supervisión estratégica y la acción táctica en terreno.

### Interfaces Conversacionales (Chatbots) en la Industria

Los *Chatbots* o asistentes virtuales son programas informáticos diseñados para simular una conversación inteligente con usuarios humanos a través de texto o voz. En el ámbito industrial, estos sistemas han evolucionado desde simples contestadores automáticos hacia herramientas operativas capaces de ejecutar flujos de trabajo complejos.

Para este proyecto, se teoriza el uso de **Bots de Mensajería** (específicamente sobre la plataforma Telegram) como una interfaz de usuario ligera. Esta elección tecnológica permite aprovechar una plataforma que ya es familiar para el operador, eliminando la curva de aprendizaje asociada a nuevas interfaces gráficas complejas. El *bot* actúa como un "facilitador digital", permitiendo reportar hallazgos o recibir alertas mediante comandos simples, transformando el paradigma de interacción tradicional: en lugar de exigir que el operador se desplace hacia una estación de trabajo fija, el "sistema va al operador" a través de su dispositivo móvil.

### Reducción de la Fricción Tecnológica

El concepto de **fricción tecnológica** se refiere a cualquier obstáculo cognitivo, físico o técnico que dificulta que un usuario complete una tarea digital. En la faena minera La Coipa, factores como el uso de guantes de seguridad, la exposición al polvo y la fatiga física aumentan significativamente esta fricción al interactuar con pantallas táctiles convencionales o formularios web extensos.

La integración de una interfaz omnicanal busca minimizar esta resistencia mediante estrategias de diseño centrado en el usuario:

* **Accesibilidad Inmediata:** Elimina la necesidad de descargar, instalar y actualizar aplicaciones nativas pesadas en dispositivos personales.
* **Interacción Asíncrona:** Permite gestionar alertas y órdenes de trabajo incluso en condiciones de conectividad intermitente, encolando los mensajes hasta recuperar la señal.
* **Agilidad de Respuesta:** Reduce la latencia operativa al notificar directamente al personal responsable, asegurando que la información predictiva generada por el motor de IA se traduzca en acciones correctivas inmediatas antes de que ocurra la falla.

FORMULACIÓN Y DELIMITACIÓN DEL PROBLEMA EN ESTUDIO

Para garantizar el éxito de una solución tecnológica industrial, es imperativo comprender a fondo el ecosistema donde esta operará. En el presente capítulo se desarrolla la caracterización detallada de la organización **SOMACOR S.A.** y el entorno operativo de la faena **La Coipa**, identificando las variables físicas y estratégicas que condicionan el proyecto.

Posteriormente, se realiza un análisis crítico de la situación actual (*As-Is*), diagramando los flujos de información manuales y detectando los puntos de "ceguera de datos" que hoy limitan la eficiencia del mantenimiento. Finalmente, sobre la base de este diagnóstico, se formalizan las necesidades del negocio y se especifican los requerimientos funcionales y no funcionales que delimitarán el alcance técnico de la plataforma CMMS propuesta.

## Descripción de la Organización

### Antecedentes de la Empresa

**SOMACOR S.A.** es una organización chilena con una consolidada trayectoria de más de cinco décadas en la prestación de servicios integrales para la gran minería e industria nacional. Su modelo de negocio se orienta a garantizar la continuidad operacional de sus clientes mediante la provisión de servicios especializados que incluyen mantenimiento industrial, aseo técnico y apoyo logístico en faenas de alta complejidad geográfica.

La empresa opera bajo un riguroso Sistema de Gestión Integrado (SGI), certificado bajo normas internacionales como la ISO 9001 (Calidad), ISO 14001 (Medio Ambiente) (International Organization for Standardization, 2015) y ISO 45001 (Seguridad y Salud Ocupacional). Estas certificaciones respaldan su compromiso con la excelencia operativa y la seguridad de sus colaboradores, factores críticos para mantener su posición como socio estratégico de compañías mineras de clase mundial, tales como **Gold Fields** en la región de Atacama.

### Misión y Visión

El direccionamiento estratégico de la organización se define a través de los siguientes lineamientos corporativos:

* **Misión:** "Proveer soluciones integrales de continuidad operacional y servicios a la minería, priorizando la seguridad de las personas, el cuidado del medio ambiente y la excelencia en la ejecución, generando valor sostenible para clientes, colaboradores y accionistas".
* **Visión:** "Consolidarse como el socio estratégico líder en servicios industriales en Chile, reconocido por su capacidad de innovación tecnológica, confiabilidad operativa y compromiso inquebrantable con los estándares de seguridad y calidad".

### Estructura Organizacional

SOMACOR S.A. sustenta sus operaciones sobre una sólida estructura organizacional diseñada para brindar servicios integrales con exigentes estándares de calidad. Esta jerarquía permite gestionar eficientemente sus diversas líneas de negocio, garantizando una supervisión efectiva y una rápida capacidad de respuesta ante los requerimientos del cliente.

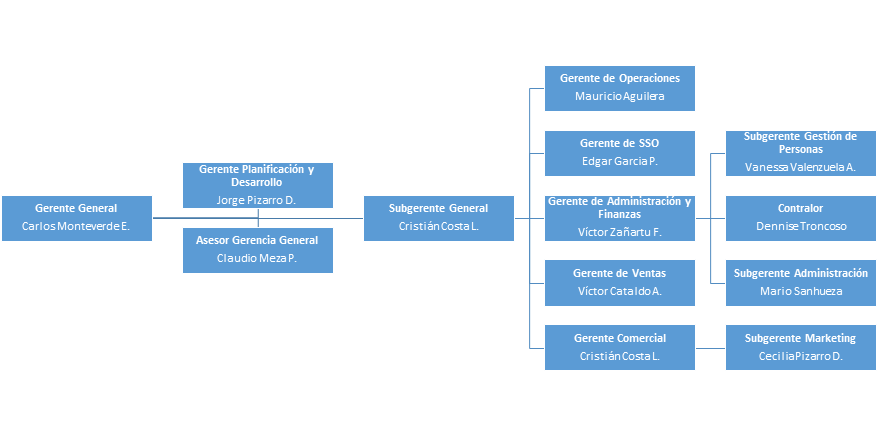


Figura 1 Organigrama Corporativo de SOMACOR S. A  
Fuente: Elaboración propia

### Localización y Entorno Operativo

El proyecto se circunscribe específicamente a las operaciones en la faena minera **La Coipa**. Este yacimiento se encuentra ubicado en la precordillera de la Región de Atacama, a una altitud superior a los 3.000 metros sobre el nivel del mar (Gold Fields, 2024).

El entorno se caracteriza por condiciones ambientales extremas que impactan directamente en la operación de los activos físicos y en la implementación de soluciones tecnológicas:

* **Oscilación Térmica:** Variaciones drásticas de temperatura que afectan la viscosidad de fluidos y fatiga de materiales.
* **Geografía Abrupta:** Terrenos irregulares y pendientes pronunciadas que exigen el máximo rendimiento de la maquinaria.
* **Polvo en Suspensión (Chuzca):** Factor ambiental que acelera el desgaste mecánico y dificulta el uso de dispositivos electrónicos sensibles en terreno.

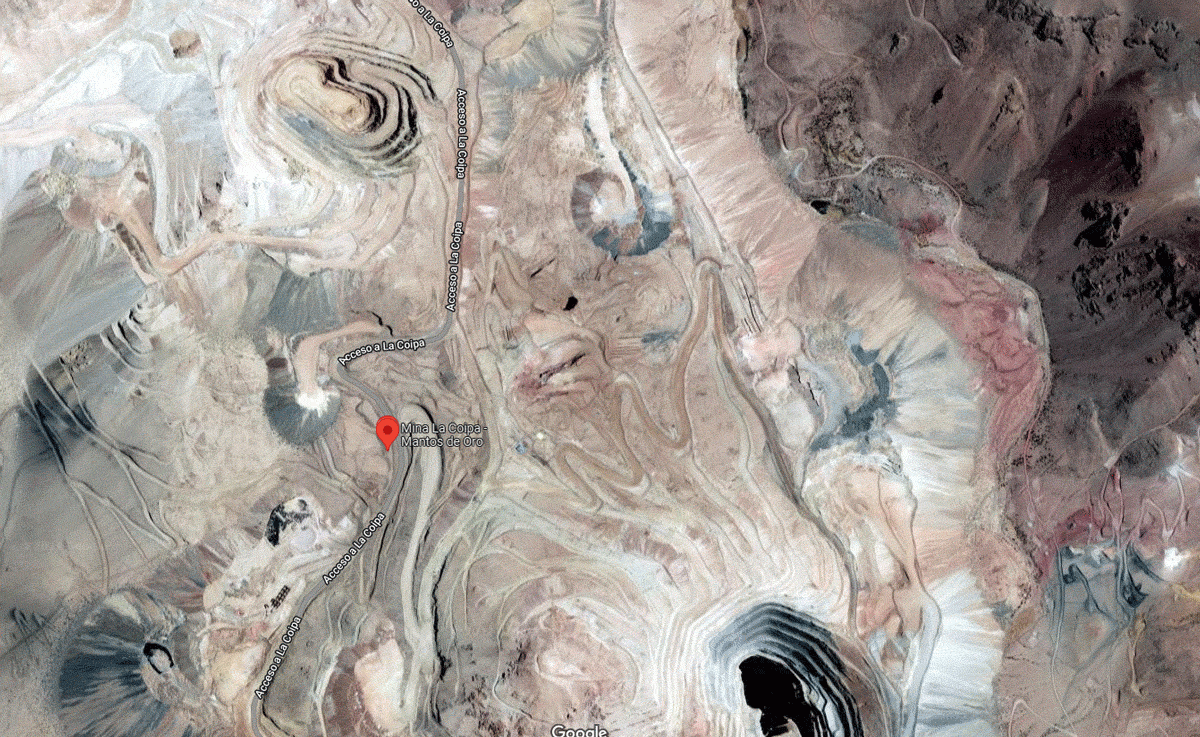


Figura 2 Ubicación geográfica de la faena La Coipa

### Infraestructura Tecnológica y Conectividad en Faena

Un factor determinante para el diseño de la solución propuesta es la capacidad instalada de Tecnologías de la Información (TI) en el lugar de operación. Actualmente, la faena presenta restricciones críticas que condicionan la arquitectura del sistema:

* **Conectividad Limitada:** La cobertura de red móvil (4G/LTE) no es homogénea en toda la faena. Existen "zonas de sombra" en los patios de operaciones y rutas de tránsito donde la señal es nula o intermitente, lo que obliga a que la solución posea capacidades de funcionamiento asíncrono.
* **Equipamiento de Usuario:** Los operadores y técnicos no cuentan con dispositivos móviles corporativos robustecidos (*rugged devices*) asignados individualmente. El acceso digital se realiza mayoritariamente a través de sus dispositivos personales (*Bring Your Own Device* - BYOD), lo que exige una solución compatible con diversos sistemas operativos y de bajo consumo de recursos.
* **Infraestructura de Servidores:** La operación no dispone de un *Data Center* local (*On-Premise*) para alojar aplicaciones complejas, dependiendo íntegramente de servicios en la nube accesibles vía internet pública.

## Análisis de la Situación Actual (As-Is)

El diagnóstico del estado actual de la gestión de activos en la faena La Coipa revela un ecosistema operativo caracterizado por una baja madurez digital y una dependencia crítica de procesos manuales. A pesar de la modernidad de la maquinaria utilizada, los flujos de información que sustentan su operación no han evolucionado al mismo ritmo, generando brechas significativas en la eficiencia y el control.

### Descripción del Proceso de Mantenimiento Actual

El ciclo de mantenimiento opera bajo un esquema predominantemente reactivo y analógico. El flujo de trabajo estándar se desarrolla en las siguientes etapas secuenciales:

1. **Inspección Pre-Uso (Checklist):** Al inicio de cada turno, el operador realiza una inspección visual del equipo y debe completar un *checklist* físico en papel. Se ha cronometrado que esta tarea consume un promedio de **10 minutos por equipo**, tiempo que se sustrae de la operación productiva efectiva.
2. **Registro de Hallazgos:** Si se detecta una anomalía (ej. nivel bajo de aceite, ruido extraño), el operador la consigna en el papel. Sin embargo, no existe un canal de transmisión inmediato; el hallazgo queda "confinado" en la bitácora física dentro de la cabina o en la caseta de supervisión.
3. **Recolección y Procesamiento Manual:** Los formularios se acumulan físicamente estimándose un volumen superior a 10 resmas mensuales y son recolectados por el supervisor de turno de manera asíncrona (al final del día o de la semana). Posteriormente, un administrativo debe transcribir manualmente estos datos a planillas de cálculo (Excel) para generar informes de estado.
4. **Toma de Decisiones:** Debido a la latencia entre la detección del síntoma en terreno y su digitalización administrativa, la gerencia carece de visibilidad en tiempo real. Esto obliga a tomar decisiones reactivas, interviniendo los equipos bajo un modelo de "mantenimiento correctivo de emergencia" solo cuando la falla funcional ya se ha manifestado.

### Diagrama de Flujo del Proceso Actual

La siguiente figura modela el flujo de información actual bajo notación BPMN (Object Management Group, 2011), evidenciando los cuellos de botella y los puntos de ruptura en la trazabilidad de la información:

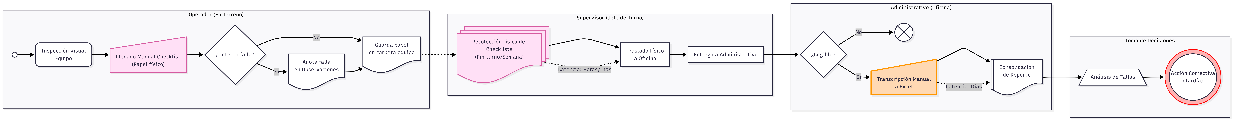


Figura 3 Diagrama flujo de proceso actual

### Identificación de Deficiencias Críticas

El análisis del proceso descrito permite identificar tres deficiencias estructurales que justifican la intervención tecnológica:

* **"Ceguera de Datos" (Data Blindness):** La información crítica sobre la salud de los activos queda dispersa en archivadores físicos sin ser procesada. Al no existir una base de datos estructurada, es imposible aplicar análisis de tendencias o modelos predictivos, transformando la gestión en una "caja negra" operativa.
* **Costos por Inactividad (Downtime):** La falta de anticipación a las fallas deriva en detenciones no programadas. Datos internos valorizan el lucro cesante de un equipo crítico (ej. *Supersucker*) en aproximadamente **$517.000 diarios**, sumado a multas contractuales que elevan la pérdida total sobre los **$710.500 por día**.
* **Falta de Trazabilidad:** No existe un historial digital unificado ("Hoja de Vida") del equipo. Esto impide auditar intervenciones pasadas, gestionar garantías de repuestos de manera efectiva y cumplir con los estándares de trazabilidad exigidos por la norma ISO 9001.

## Necesidades del Negocio y Requerimientos del Sistema

Sobre la base del análisis de la situación actual y las deficiencias detectadas en la gestión manual, se han formalizado las necesidades del negocio que la solución tecnológica debe satisfacer. Estas necesidades se traducen técnicamente en Requerimientos Funcionales (lo que el sistema debe hacer) y No Funcionales (cómo debe comportarse el sistema).

### tabla de Requerimientos Funcionales Clave

Tabla 1 Requerimientos Funcionales Clave

|  |  |  |  |
| --- | --- | --- | --- |
| Código | Requerimiento | Descripción | Prioridad |
| RF-01 | Autenticación y RBAC | El sistema debe permitir el inicio de sesión seguro mediante tokens JWT y gestionar roles (Operador, Supervisor, Administrador) para restringir el acceso a funciones críticas. | Alta |
| RF-03 | Gestión de Hoja de Vida | El sistema debe permitir registrar y actualizar la ficha técnica completa de cada activo (marca, modelo, horómetro), actuando como la fuente de verdad para la trazabilidad histórica. | Alta |
| RF-05 | Gestión de Órdenes de Trabajo (OT) | El sistema debe permitir la creación, asignación y seguimiento de OTs correctivas, preventivas y predictivas, registrando tiempos y responsables. | Alta |
| RF-12 | Bitácora Digital (Checklists) | El sistema debe permitir la digitalización de pautas de inspección diaria, capturando hallazgos cualitativos que alimentarán el historial del equipo. | Alta |
| RF-13 | Motor de Inferencia Predictiva | El sistema debe ejecutar periódicamente el algoritmo *Random Forest* para calcular la probabilidad de fallo de cada equipo en una ventana de tiempo futura (ej. 7 días). | Alta |
| RF-15 | Gatillado Automático de OT | El sistema debe generar automáticamente una propuesta de Orden de Trabajo cuando la probabilidad de fallo supere el umbral crítico definido (ej. >85%). | Media |
| RF-19 | Dashboard de KPIs | El sistema debe visualizar indicadores en tiempo real como Disponibilidad Física, MTBF y MTTR, facilitando la toma de decisiones gerenciales. | Media |
| RF-23 | Notificaciones Omnicanal | El sistema debe enviar alertas automáticas (Push) a los supervisores a través de Telegram ante eventos críticos, como una nueva predicción de alto riesgo. | Alta |
| RF-24 | Consulta Remota vía Bot | El Bot de Telegram debe responder a comandos directos (ej. /status, /ot) para entregar información del estado de la flota sin necesidad de acceder a la web. | Media |

### tabla de Requerimientos Funcionales Clave

Tabla 2 Requerimientos No Funcionales Clave

|  |  |  |  |
| --- | --- | --- | --- |
| Código | Requerimiento | Descripción | Criterio de Aceptación |
| RNF-01 | Seguridad en Autenticación | Implementación de tokens JWT para la web y un protocolo de vinculación por código temporal para Telegram, evitando el traspaso de credenciales por texto plano. | Estándar OAuth2 / JWT. |
| RNF-02 | Rendimiento de API | La API REST debe responder a las solicitudes de transacción estándar en menos de 200ms para asegurar una experiencia fluida en redes móviles. | Latencia < 200ms (p95). |
| RNF-03 | Disponibilidad y Resiliencia | La arquitectura debe ser capaz de desacoplar el procesamiento intensivo de datos (entrenamiento de IA) del servidor web para evitar la degradación del servicio. | Uptime > 99.5%. |
| RNF-04 | Escalabilidad Horizontal | La infraestructura *Headless* debe soportar el crecimiento en el volumen de datos y usuarios sin requerir cambios estructurales en el código base. | Arquitectura basada en microservicios/contenedores. |
| RNF-05 | Usabilidad Móvil | La interfaz web debe ser responsiva y el Bot de Telegram debe operar con un bajo consumo de datos, optimizado para la red 4G intermitente de la faena. | Diseño *Mobile-First*. |
| RNF-08 | Precisión del Modelo | El modelo predictivo inicial (basado en datos sintéticos) debe alcanzar una Sensibilidad (*Recall*) superior al 85% para minimizar los Falsos Negativos (fallas no detectadas). | Recall > 85% en validación. |

METODOLOGÍA DE TRABAJO Y PLANIFICACIÓN

La naturaleza innovadora de este proyecto, que combina el desarrollo de software industrial con la incertidumbre inherente al entrenamiento de modelos de Inteligencia Artificial (especialmente en un escenario de *Cold Start*), exige un enfoque de gestión flexible y adaptativo. A diferencia de los modelos lineales tradicionales, que requieren una definición rígida de requisitos al inicio, este proyecto necesita un marco que permita la validación temprana de hipótesis y la incorporación ágil de cambios basados en el *feedback* operativo.

Por estas razones, se ha seleccionado la metodología ágil **Scrum** como el marco de trabajo para la gestión del proyecto. En el presente capítulo se fundamenta esta elección técnica, se definen los roles y responsabilidades del equipo de trabajo, y se detalla el cronograma de actividades estructurado en iteraciones (*Sprints*) para asegurar el cumplimiento de los objetivos en los plazos académicos y operativos establecidos.

## Selección de la Metodología Ágil (Scrum)

Para la gestión y desarrollo del sistema CMMS predictivo, se optó por la metodología **Scrum**. Esta elección se fundamenta en la naturaleza evolutiva del proyecto, donde los requerimientos de la operación minera pueden variar dinámicamente y la validación de los modelos predictivos requiere ciclos rápidos de retroalimentación y ajuste.

A diferencia de los modelos tradicionales en cascada (*Waterfall*), que exigen una planificación exhaustiva y rígida antes de escribir código, Scrum permite organizar el trabajo en bloques temporales cortos y fijos denominados **Sprints** (en este proyecto, de 2 semanas de duración) (Schwaber & Sutherland, 2020).

Cada Sprint tiene como objetivo entregar un incremento de software funcional y potencialmente desplegable, lo que facilita la detección temprana de errores y la adaptación continua a las necesidades del usuario final en la faena La Coipa.

### Justificación de la Elección

La adopción de Scrum para este proyecto responde a tres factores técnicos y estratégicos críticos:

1. **Flexibilidad ante la Incertidumbre (Gestión del *Cold Start*):** El entrenamiento del modelo *Random Forest* en un escenario de escasez de datos no es un proceso lineal. Requiere iterar rápidamente sobre la generación de datos sintéticos, ajustar hiperparámetros y validar resultados. Scrum permite realizar estas adaptaciones experimentales en ciclos cortos sin comprometer la estructura general del desarrollo.
2. **Entrega Temprana de Valor (MVP):** La metodología prioriza el desarrollo de las funcionalidades más críticas (*Core*), como el registro de activos y el Bot de Telegram. Esto permite que la organización comience a percibir valor operativo desde las primeras etapas (Sprints 3-4), validando la utilidad de la herramienta en terreno antes de su finalización total.
3. **Colaboración y Transparencia:** Scrum fomenta una comunicación fluida entre el equipo de desarrollo y los interesados (*stakeholders*) de SOMACOR S.A. mediante ceremonias regulares (Revisiones de Sprint). Esto asegura que el producto final no sea una "caja negra" tecnológica, sino una solución alineada con las expectativas reales de la Gerencia de Operaciones.

## Definición de Roles y Responsabilidades

En concordancia con el marco de trabajo Scrum, se han definido roles claros para garantizar la gobernanza del proyecto y la correcta ejecución técnica (Schwaber & Sutherland, 2020). La distribución de responsabilidades se estructuró para maximizar la autonomía del equipo de desarrollo y asegurar la alineación estratégica con los objetivos de SOMACOR S.A. en la faena La Coipa.

### Product Owner (Dueño del Producto)

Este rol actúa como el representante de la voz del cliente y de los interesados (*stakeholders*). Su responsabilidad principal es maximizar el valor del producto, gestionando el *Product Backlog* (lista de requerimientos) y priorizando las funcionalidades críticas, como el módulo predictivo y las alertas omnicanal. Es el encargado de validar que los entregables cumplan con los criterios de aceptación operativos antes del despliegue.

### Scrum Master (Facilitador)

El *Scrum Master* es responsable de asegurar que el equipo se adhiera a los principios y prácticas ágiles. Su función es eliminar impedimentos técnicos o administrativos que puedan bloquear el avance del desarrollo (por ejemplo, problemas de configuración en la nube o validación de datos). Actúa como un líder servicial que protege al equipo de interrupciones externas y facilita las ceremonias de Scrum (reuniones de planificación, revisión y retrospectiva).

### Equipo de Desarrollo (Development Team)

Conformado por los autores del proyecto, es un equipo multidisciplinario y autogestionado encargado de la construcción del incremento de software "Terminado". Este equipo posee todas las habilidades necesarias para convertir los elementos del *Backlog* en funcionalidades operativas, abarcando desde el diseño de la arquitectura *Backend* en Django hasta la implementación de la interfaz *Frontend* en React y el entrenamiento de los modelos de IA.

Tabla 3 Roles y Responsabilidades del Proyecto

|  |  |  |
| --- | --- | --- |
| Rol | Encargado | Responsabilidades Específicas |
| Product Owner | Por definir | Definir la visión del producto alineada con SOMACOR. Aprobar o rechazar los entregables al final de cada Sprint. Priorizar las historias de usuario según el valor de negocio. |
| Scrum Master | Lucas Gallardo | Gestionar el tablero Kanban en Jira/Trello. Eliminar bloqueos técnicos en el despliegue a la nube. Asegurar el cumplimiento del cronograma de Sprints. |
| Development Team | Matías Morales | Líder de Backend e Infraestructura: Desarrollo de la API RESTful en Django. Configuración de servicios en la nube (Railway/PostgreSQL). Implementación del Bot de Telegram.  ingeniería de datos y entrenamiento del modelo Random Forest. |
| Development Team | Lucas Gallardo | Líder de Frontend: Desarrollo de la interfaz de usuario en React JS.  Diseño de la experiencia de usuario (UX/UI). |

## Planificación y Cronograma de Trabajo

La ejecución del proyecto se estructuró en un horizonte temporal planificado, dividido en iteraciones (*Sprints*) que permiten una evolución controlada del producto. El cronograma se articula en **cinco fases estratégicas**, cubriendo desde el levantamiento inicial hasta el cierre del proyecto.

### Fase 1: Análisis y Definición de Requerimientos (Sprint 1)

Etapa inicial enfocada en comprender el ecosistema operativo de la faena y definir el alcance técnico.

* **Actividades Clave:** Levantamiento de procesos manuales (*As-Is*), definición detallada de historias de usuario en el *Backlog*, y formalización de la estrategia de datos para mitigar el problema del *Cold Start*.
* **Entregable:** Documento de Especificación de Requisitos y *Product Backlog* priorizado.

### Fase 2: Diseño de Arquitectura y Datos (Sprint 2)

Fase dedicada a la ingeniería de la solución antes de la codificación masiva.

* **Actividades Clave:** Diseño del Modelo de Base de Datos (DER) en PostgreSQL, definición de la arquitectura *Headless* (API REST), y diseño de los algoritmos para la generación de datos sintéticos necesarios para el entrenamiento inicial de la IA.
* **Entregable:** Diagramas de arquitectura aprobados y scripts de generación de datos listos.

### Fase 3: Construcción y Desarrollo (Sprints 3-4)

Etapa de ejecución intensiva donde se materializa la solución tecnológica.

* **Actividades Clave:**
  + **Backend:** Desarrollo de la API en Django y configuración de servicios en la nube (*Railway*).
  + **IA:** Entrenamiento del modelo *Random Forest* v1.0 utilizando los datos sintéticos generados.
  + **Frontend/Bot:** Implementación de la interfaz web en React y programación de los comandos del Bot de Telegram.
  + **Entregable:** MVP (*Producto Mínimo Viable*) funcional con capacidad de predicción y registro.

### Fase 4: Pruebas y Aseguramiento de Calidad (Sprint 5)

Fase crítica para validar la robustez y precisión del sistema.

* **Actividades Clave:** Ejecución de pruebas integrales (*End-to-End*), validación de métricas de desempeño del modelo (buscando Sensibilidad > 85%), y pruebas de usabilidad del Bot con usuarios clave para asegurar la baja fricción tecnológica.
* **Entregable:** Informe de QA y versión estable del software.

### Fase 5: Implementación y Cierre (Sprint 6)

Etapa final de despliegue y formalización del proyecto.

* **Actividades Clave:** Puesta en marcha en ambiente productivo, capacitación a operadores y supervisores, elaboración de manuales de usuario y redacción final de la Memoria de Título.
* **Entregable:** Sistema operando en producción y proyecto finalizado.

Captura de pantalla de computadora

El contenido generado por IA puede ser incorrecto.

Figura 4 Carta Gantt del Proyecto

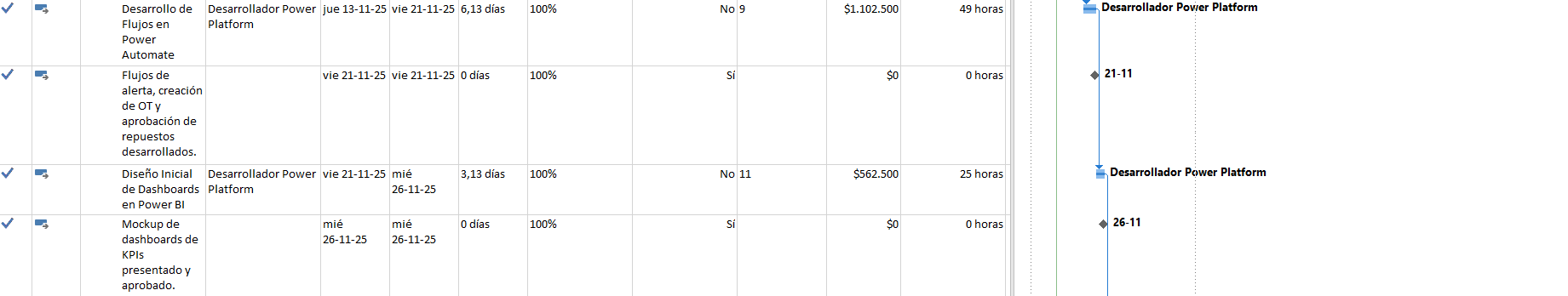
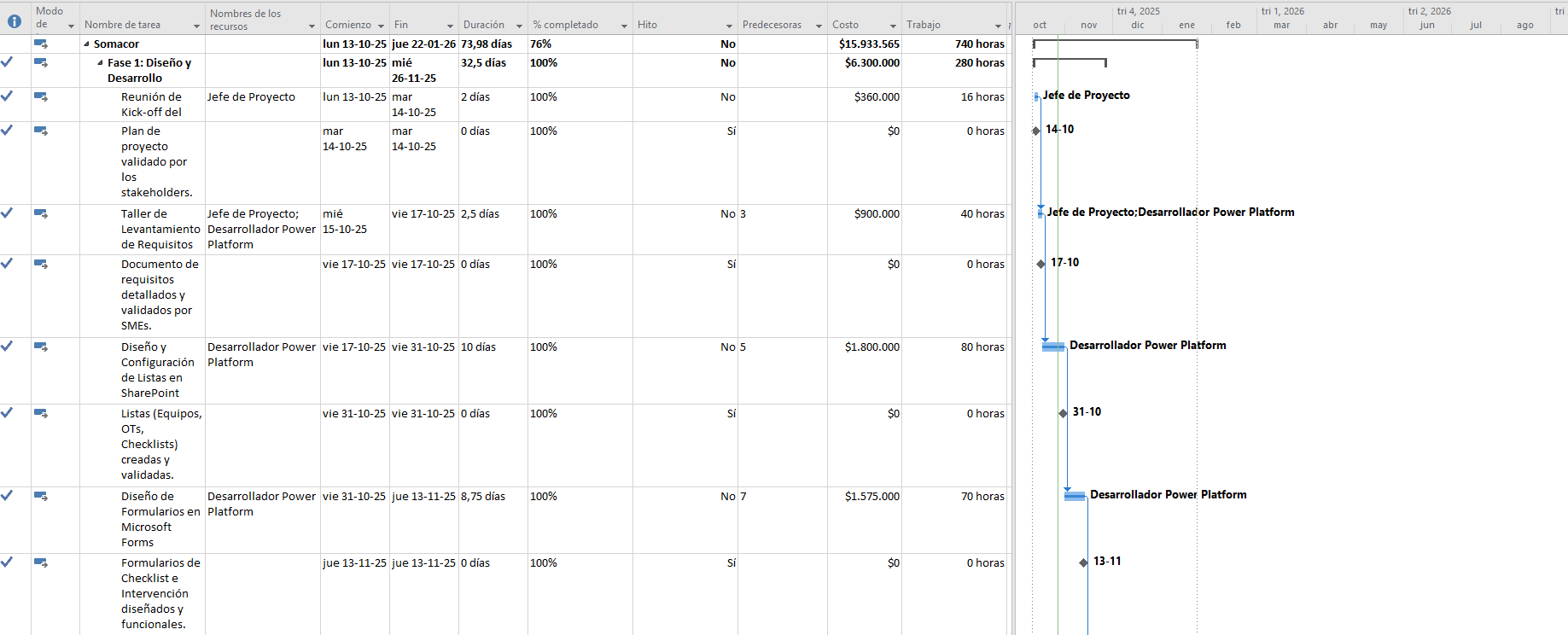


Figura 5 Carta Gantt del proyecto Fase 1

Imagen que contiene interior, computer, cubierto, llenado

El contenido generado por IA puede ser incorrecto.

Figura 6 Carta Gantt del proyecto Fase 2

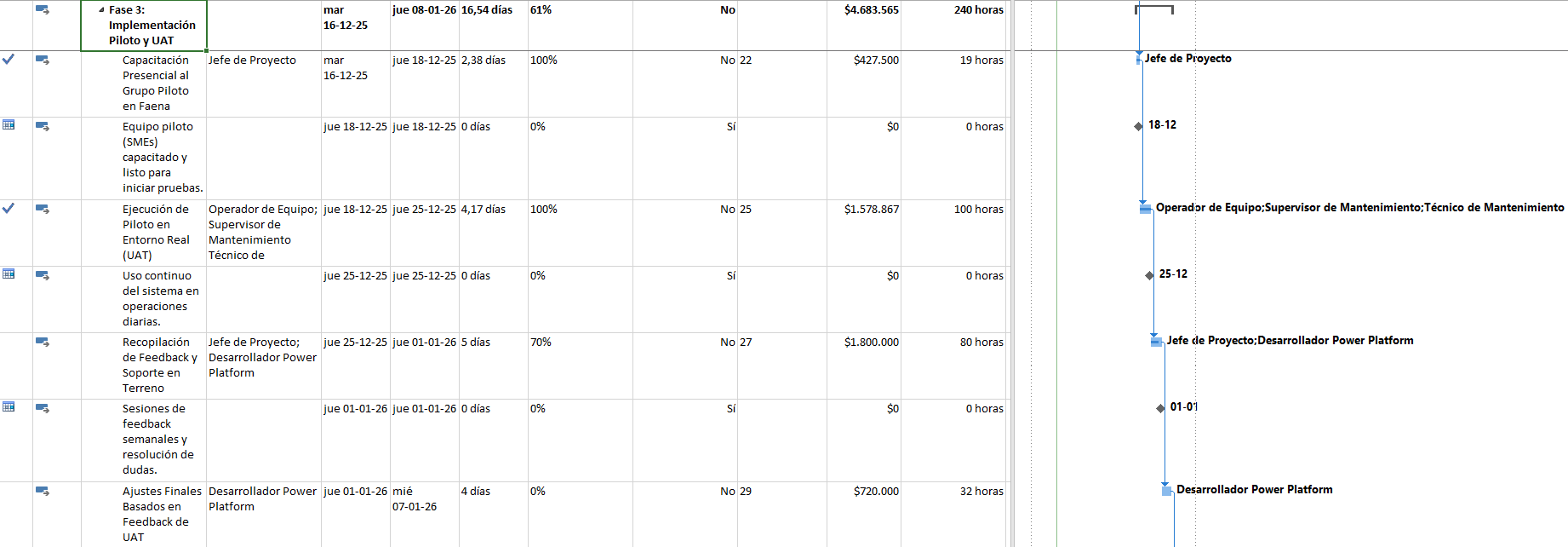
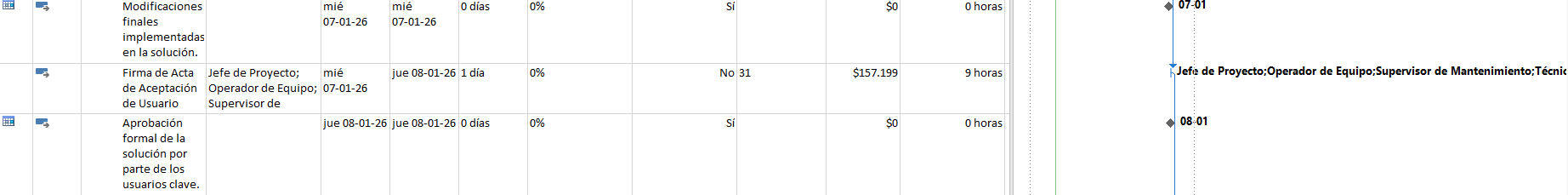


Figura 7 Carta Gantt del proyecto Fase 3

Imagen que contiene interior, mapa, computadora, computer

El contenido generado por IA puede ser incorrecto.

Figura 8 Carta Gantt del proyecto Fase 4

Imagen que contiene interior, mapa, computadora, tabla

El contenido generado por IA puede ser incorrecto.

Figura 9 Carta Gantt del proyecto Fase 5

EVALUACIÓN DE FACTIBILIDAD

Para asegurar el éxito y la sostenibilidad de un proyecto de ingeniería, no basta con demostrar su funcionalidad técnica; es indispensable validar su viabilidad desde una perspectiva integral de negocio. En este capítulo, se somete la propuesta del "Sistema de Gestión de Mantenimiento Computarizado (CMMS) Predictivo" a un riguroso análisis de factibilidad multidimensional.

El estudio abarca cuatro ejes críticos: la **Factibilidad Técnica**, que valida la disponibilidad de los recursos de *hardware* y *software* necesarios para la implementación; la **Factibilidad Económica**, que contrasta la inversión inicial y los costos operativos frente a los beneficios financieros proyectados, utilizando indicadores como el VAN y la TIR; la **Factibilidad Legal**, que asegura el cumplimiento normativo vigente; y la **Factibilidad Operacional**, que evalúa la capacidad de la organización para adoptar la nueva tecnología sin disrupciones en la faena La Coipa.

## Factibilidad Técnica

La evaluación de factibilidad técnica determina si la organización dispone o puede acceder a los recursos tecnológicos necesarios para desarrollar, implementar y sostener el sistema propuesto. Dado que el proyecto se basa en una arquitectura de *software* moderna en la nube, la viabilidad no depende de la adquisición de servidores físicos costosos, sino de la selección adecuada del *stack* tecnológico y la compatibilidad con el equipamiento existente.

### Tecnologías de Desarrollo y Librerías (*Stack*)

Para la construcción de la solución se han seleccionado herramientas de código abierto (*Open Source*) y amplia comunidad, garantizando la sostenibilidad y escalabilidad del proyecto sin costos de licenciamiento prohibitivos.

* **Lenguajes de Programación:** Python 3.10+ (Backend y Lógica de IA) y JavaScript ES6+ (Frontend).
* **Frameworks:** Django 4.2 (API REST) y React 18 (Interfaz de Usuario).
* **Ciencia de Datos:** Librerías *Scikit-learn* y *Pandas* para la implementación del algoritmo *Random Forest* y el procesamiento de datos sintéticos.
* **Control de Versiones:** Git y GitHub para la gestión del código fuente y trabajo colaborativo.

### Requisitos de Infraestructura y Despliegue (*Cloud*)

El sistema opera bajo un modelo de "Plataforma como Servicio" (PaaS), lo que elimina la necesidad de mantenimiento de hardware físico en la faena.

Tabla 4 Requerimientos de Infraestructura en la Nube

|  |  |  |
| --- | --- | --- |
| Componente | Servicio Seleccionado | Especificaciones Técnicas Requeridas |
| Servidor de Aplicaciones | Railway / Render | Contenedor Docker con soporte para Python/Django. Mínimo 1 vCPU y 512 MB RAM para ambiente de producción. |
| Hosting Frontend | Vercel | Servicio de CDN global para entrega de contenido estático (React SPA) con baja latencia. |
| Base de Datos | PostgreSQL (Gestionado) | Instancia de base de datos relacional con soporte para conexiones concurrentes y backups automáticos diarios. |
| Cola de Tareas | Redis | Servicio de broker de mensajes para gestionar el procesamiento asíncrono del modelo predictivo. |

### Requisitos del Usuario Final

Considerando las restricciones de conectividad y equipamiento en la faena La Coipa, la solución ha sido optimizada para funcionar en dispositivos de gama media y redes inestables.

* **Dispositivos Móviles:** *Smartphones* o *Tablets* con sistema operativo Android (versión 8.0 o superior) o iOS (versión 13 o superior). No se requiere hardware robustecido específico, permitiendo el modelo BYOD (*Bring Your Own Device*).
* **Navegador Web:** Google Chrome, Firefox o Safari (versiones actualizadas) para el acceso al Dashboard administrativo.
* **Aplicación de Mensajería:** Tener instalada la aplicación Telegram para la interacción con el Bot Omnicanal.
* **Conectividad:** Acceso a red de datos móvil (3G/4G) o Wi-Fi corporativo. El sistema tolera desconexiones temporales mediante encolado de mensajes en el Bot.

## Factibilidad Económica

La evaluación económica se ha estructurado bajo un modelo de negocio de **Prestación de Servicios Tecnológicos (SaaS)**. En lugar de evaluar el proyecto meramente como un centro de costos interno, se analiza como una unidad de negocio capaz de generar rentabilidad mediante la venta de licencias y servicios de desarrollo evolutivo ("Bolsa de Horas") a la faena La Coipa y potenciales nuevos clientes mineros.

El horizonte de evaluación se establece en **24 meses (2 años)**, alineado con la duración estándar de los contratos de servicio en la minería.

### Estructura de Inversión Inicial (CAPEX)

Corresponde al capital necesario para constituir la empresa proveedora, adquirir el equipamiento de desarrollo y financiar la construcción del MVP (*Producto Mínimo Viable*) antes de la primera venta.

Tabla 5 Inversión Inicial (Puesta en Marcha)

|  |  |  |
| --- | --- | --- |
| Ítem de Inversión | Detalle | Monto (CLP) |
| Equipamiento Tecnológico | 2 notebooks Alto Rendimiento + Periféricos. | $2.100.000 |
| Constitución y Legal | Escritura empresa, dominio web, registro de marca. | $650.000 |
| Desarrollo del MVP | HH invertidas en la fase de Tesis (Capitalizado). | $13.372.156 |
| TOTAL, INVERSIÓN | Costo total de arranque. | $16.122.156 |

### Presupuesto Operacional (Ingresos y Egresos)

El modelo de operación se basa en un esquema de **Ingresos Recurrentes Mensuales (MRR)** que superan a los costos fijos, generando un margen de contribución positivo desde el primer mes de ventas.

Tabla 6 Estructura Mensual de Ingresos y Egresos

|  |  |  |
| --- | --- | --- |
| **Concepto** | **Detalle** | **Monto Mensual (CLP)** |
| **(+) INGRESOS** | **Facturación Total Proyectada** | **$4.950.000** |
| \* Licenciamiento SaaS | 3 Faenas (La Coipa + 2 Expansiones) | $1.050.000 |
| \* Soporte y Mantención | Fee fijo por disponibilidad 24/7 | $500.000 |
| \* Bolsa de Horas | 400 Horas de Desarrollo Evolutivo | $3.400.000 |
| **(-) EGRESOS** | **Costo Operativo Total (OPEX)** | **$2.600.000** |
| \* Infraestructura Cloud | GCP (Cloud Run + SQL) + Firebase | $250.000 |
| \* Equipo de Desarrollo | RRHH (2 Desarrolladores Junior/Mid) | $2.200.000 |
| \* Gastos Admin. | Contabilidad, Servicios Básicos, Internet | $150.000 |
| **(=) FLUJO NETO** | **Margen Operacional Mensual** | **$2.350.000** |

### Flujo de Caja Proyectado

A continuación, se presenta el resumen del flujo de caja a 24 meses. Se observa que el proyecto, al tener una estructura de costos liviana (sin deuda bancaria y con infraestructura en la nube), genera flujos positivos constantes tras la inversión inicial.

Tabla 7 Resumen del Flujo de Caja (en CLP)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Periodo** | **Año 0 (Inversión)** | **Mes 1 (Inicio)** | **Mes 12 (Año 1)** | **Mes 24 (Año 2)** |
| **Total Ingresos** | $0 | $4.950.000 | $4.950.000 | $4.950.000 |
| **Total Egresos** | $16.122.156 | $2.600.000 | $2.600.000 | $2.600.000 |
| **Flujo Neto** | **-$16.122.156** | **$2.350.000** | **$2.350.000** | **$2.350.000** |
| **Flujo Acumulado** | **-$16.122.156** | -$13.772.156 | $12.077.844 | $40.277.844 |

(Ver Anexo para el detalle completo mes a mes).

### Indicadores de Rentabilidad y Análisis Financiero

La evaluación financiera arroja resultados altamente positivos, característicos de proyectos de tecnología escalables donde la inversión en activos físicos es baja en comparación con el potencial de facturación.

|  |  |  |
| --- | --- | --- |
| **Indicador** | **Valor** | **Interpretación** |
| **VAN (Valor Actual Neto)** | **$35.024.578** | Calculado con una tasa de descuento anual del 10%. El proyecto genera una riqueza neta superior a los 35 millones de pesos al cabo de 2 años. |
| **TIR (Tasa Interna de Retorno)** | **14% Mensual** | La rentabilidad mensual del dinero invertido es del 14% (equivalente a >300% anual), lo que valida la eficiencia del capital en este modelo de negocio. |
| **Payback (Recuperación)** | **7 Meses** | El punto de equilibrio (*Break-even*) se alcanza en el mes 7, momento en el cual los flujos acumulados cubren totalmente la inversión inicial de 16 millones. |

Tabla 8 Indicadores de Viabilidad Económica

**Análisis de los Resultados:** La proyección financiera demuestra que el proyecto es robusto y sostenible. El **Payback de 7 meses** es un indicador clave de bajo riesgo, ya que la inversión se recupera en menos de un tercio del horizonte total del proyecto. Además, el **VAN positivo** confirma que, incluso descontando el valor del dinero en el tiempo, la implementación del CMMS genera un valor económico sustancial tanto para los desarrolladores (como proveedores) como para la mandante (mediante la continuidad operativa asegurada).

### Justificación del Modelo de Negocio Propuesto

La viabilidad económica del proyecto se sustenta no solo en los indicadores financieros positivos (VAN y TIR), sino en la robustez estructural del modelo de negocio **SaaS (*Software as a Service*)** adoptado. A diferencia de los proyectos de inversión tradicionales en minería, que requieren alta inyección de capital en activos físicos (*Heavy Asset*), esta solución tecnológica presenta ventajas comparativas que justifican su alta rentabilidad:

1. **Escalabilidad con Costo Marginal Tendiente a Cero:** Una vez desarrollado el MVP (*Producto Mínimo Viable*), el costo de agregar un nuevo cliente (una nueva faena) es insignificante en comparación con los ingresos que genera. La infraestructura en la nube permite replicar el servicio sin necesidad de reescribir el código, lo que explica por qué la Tasa Interna de Retorno (TIR) mensual del **14%** supera ampliamente los márgenes tradicionales de la industria.
2. **Ingresos Recurrentes y Previsibilidad (MRR):** El modelo de cobro por licencia mensual y mantenimiento asegura un flujo de caja constante y predecible (*Monthly Recurring Revenue*). Esto reduce la volatilidad financiera y facilita la planificación de inversiones futuras, a diferencia de los modelos de "venta única" (*One-off*) donde se depende constantemente de cerrar nuevos negocios para sobrevivir.
3. **Valorización del Activo Intangible:** Más allá del flujo de caja inmediato, el desarrollo del software genera un activo intelectual valorizable para la empresa. La propiedad del código fuente y los algoritmos de Inteligencia Artificial entrenados constituyen un patrimonio tecnológico que aumenta el valor comercial de SOMACOR S.A., posicionándola no solo como una empresa de servicios, sino como un actor de innovación tecnológica (*Tech-enabled Service*).

## Factibilidad Legal

Este análisis verifica que el proyecto cumpla con el marco normativo vigente en Chile, tanto en aspectos laborales como en la protección de la propiedad intelectual y la seguridad de la información.

### Normativa Laboral y Seguridad

El sistema se alinea con el **Artículo 184 del Código del Trabajo** (Ministerio del Trabajo y Previsión Social, 2024), que obliga al empleador a tomar todas las medidas necesarias para proteger eficazmente la vida y salud de los trabajadores. Al implementar una herramienta predictiva que anticipa fallas mecánicas catastróficas, SOMACOR S.A. refuerza el cumplimiento de esta obligación legal, reduciendo la exposición al riesgo de sus trabajadores en faena.

### Propiedad Intelectual y Licenciamiento

El desarrollo del software respeta los derechos de autor y licencias vigentes:

* **Software Propio:** El código fuente desarrollado (Frontend y Backend) es propiedad intelectual de SOMACOR S.A. como activo intangible de la empresa.
* **Librerías de Terceros:** Se utilizan exclusivamente librerías y *frameworks* bajo licencias de código abierto permisivas (MIT, BSD, Apache 2.0) (Open Source Initiative, 2024), como React y Django, lo que permite su uso comercial sin incurrir en regalías o infracciones legales.
* **Protección de Datos:** El sistema cumple con la **Ley N° 19.628 sobre Protección de la Vida Privada** (Ministerio de Justicia, 1999), implementando mecanismos de encriptación para proteger los datos personales de los operadores (RUT, nombres) almacenados en la base de datos.

## Factibilidad Operacional

La factibilidad operacional evalúa la capacidad de la organización para adoptar, utilizar y sostener el sistema en el tiempo, considerando los factores humanos y culturales.

### Gestión del Cambio y Adopción

Se ha identificado una potencial "resistencia al cambio" por parte de operadores acostumbrados a procesos manuales. Para mitigar esto, la solución incorpora estrategias de diseño centradas en la usabilidad:

* **Baja Fricción:** El uso de un **Bot de Telegram** elimina la necesidad de aprender a usar una nueva aplicación compleja. La interfaz es familiar (chat) y los comandos son intuitivos.
* **Valor Percibido:** El sistema agiliza el trabajo del operador (menos escritura manual), lo que incentiva su uso voluntario.

### Disponibilidad del Recurso Humano

La empresa cuenta con el personal necesario para operar el sistema. No se requiere contratar nuevos perfiles especializados en faena; los roles actuales (Supervisores y Planificadores) asumirán la administración funcional del *dashboard* como parte de la optimización de sus tareas actuales, sustituyendo el tiempo dedicado a la transcripción de papel por tiempo de análisis de datos.

## Gestión de Riesgos y Plan de Contingencia

La gestión de riesgos es un proceso continuo y proactivo orientado a identificar, analizar y responder a los factores que podrían afectar negativamente el éxito del proyecto.

### Identificación de Riesgos

A continuación, se listan algunos de los principales riesgos que pueden ralentizar el progreso del proyecto

**Riesgo Identificado:**

* Adopción insuficiente del **CMMS** por parte de los usuarios en faena
* Baja calidad de la *data entry* (inconsistencia, duplicados, registros incompletos)
* Fallas en aplicación web con capacidades offline (PWA) o mala conectividad en faena
* Limitaciones de *PostgreSQL* (*Cloud SQL*) / Django *REST Framework* (límites de ejecución o volumen)
* Errores en predicciones del modelo IA (falsos positivos/negativos)
* Dependencia del ecosistema Google (vendor lock-in)
* Integraciones incorrectas entre **Reac**t con *TypeScript*, Django *REST* ***Framework*** y ***Cloud Run***
* Brechas de seguridad o permisos mal configurados
* Escalamiento futuro a ***Vertex AI*** no compatible o costoso
* Falta de reentrenamientos periódicos del modelo IA

Estos riesgos se utilizarán para hacer la matriz de riegos, donde se clasificará el riesgo por su valorización, impactos, probabilidad y él nivel de riesgo correspondiente.

### Matriz de riesgo

Tabla 9 Criterios de evaluación

|  |  |
| --- | --- |
| Criterio | Descripción |
| valorización | Se toma como criterio de evaluación la valorización que le daría la empresa en caso de ocurrir dicho evento. |
| Impacto | Se tomará como criterio de evaluación el impacto que generaría a la empresa en caso de fallar el componente físico. |
| Probabilidad | Se toma como criterio de evaluación que tan probable es que ocurra dicho evento. |
| Nivel de Riesgo | Se calcula el nivel de riesgo como: (Valorización + (Impacto\*Probabilidad)). Mientras más alto, más riesgo significa para la empresa. |

|  |  |
| --- | --- |
| **Valorización** | **Nivel** |
| Muy Alta | 5 |
| Alta | 4 |
| Media | 3 |
| Baja | 2 |
| Muy Baja | 1 |

Junto a cada criterio, se asignan distintos niveles para la evaluación de cada uno, siendo los valores mayores para identificar los de mayor valorización, impacto y probabilidad de ocurrencia respectivamente; se les asignan colores a los distintos niveles para una mejor apreciación y contraste de sus valores. Estos son:

|  |  |
| --- | --- |
| **Probabilidad** | **Nivel** |
| Alta | > 0,5 |
| Media | 0,5 |
| Baja | < 0,5 |

|  |  |
| --- | --- |
| **Impacto** | **Nivel** |
| Critico | 2 |
| Considerado | 1 |
| Ninguno | 0 |

Tabla 10 Niveles de los criterios de evaluación

Finalmente, en base a lo anterior se procede al desarrollo de la Matriz de riesgos:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Riesgos Identificados | Valoración | | Impacto | | Probabilidad | | Nivel de Riesgo |
| Adopción insuficiente del CMMS por parte de los usuarios en faena | Alta | 4 | Critico | 2 | Media | 0,2 | 4,4 |
| Baja calidad de la *data entry* (inconsistencia, duplicados, registros incompletos) | Muy Alta | 5 | Critico | 2 | Alta | 0,3 | 5,6 |
| Fallas en aplicación web con capacidades offline (PWA) o mala conectividad en faena | Media | 3 | Considerado | 1 | Media | 0,2 | 3.2 |
| Limitaciones de PostgreSQL (Cloud SQL) / *Django REST Framework* (límites de ejecución o volumen) | Baja | 2 | Considerado | 1 | Baja | 0,1 | 2,1 |

Tabla 11 Matriz de riesgos

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Errores en predicciones del modelo IA (falsos positivos/negativos) | Alta | 4 | Critico | 2 | Media | 0,2 | 4,4 |
| Dependencia del ecosistema Google | Baja | 2 | Considerado | 1 | Baja | 0,1 | 2,1 |
| |  | | --- | |  |   Integraciones incorrectas entre *React* con *TypeScript,* *Django REST Framework* y *Cloud Run* | Alta | 4 | Critico | 2 | Media | 0,2 | 4,4 |
| Brechas de seguridad o permisos mal configurados | Media | 3 | Critico | 2 | Baja | 0,1 | 3,2 |
| |  | | --- | |  |   Escalamiento futuro a *Vertex AI* no compatible | Baja | 2 | Considerado | 1 | Baja | 0,1 | 2,1 |
| Falta de reentrenamientos del modelo IA | Muy Alta | 5 | Critico | 2 | Alta | 0,3 | 5,6 |

### Análisis Cualitativo

La siguiente tabla evalúa los aspectos cualitativos de la solución propuesta frente a un paquete de software CMMS estándar.

|  |  |  |
| --- | --- | --- |
| **Impacto** | | |
| Bajo | Medio | Alto |
| **Probabilidad** | Baja | Trivial | Tolerable | Moderado |
|  | Media | Tolerable | Moderado | Significativo |
|  | Alta | Moderado | Significativo | Critico |

Tabla 12 Criterios del análisis cualitativo

|  |  |  |  |
| --- | --- | --- | --- |
| Riesgos Identificados | Impacto | Probabilidad | Prioridad |
| Adopción insuficiente del CMMS por parte de los usuarios en faena | Alta | Medio | Significativo |
| Baja calidad del *data entry* (inconsistencia, duplicados, registros incompletos) | Alta | Medio | Significativo |
| Fallas en aplicación web con capacidades *offline* (PWA) o mala conectividad en faena | Medio | Baja | Tolerable |
| Limitaciones de *PostgreSQL* (Cloud SQL) / *Django REST Framework* (límites de ejecución o volumen) | Medio | Medio | Moderado |

|  |  |  |  |
| --- | --- | --- | --- |
| Errores en predicciones del modelo IA (falsos positivos/negativos) | Alta | Alta | Critico |
| Dependencia del ecosistema Google (*vendor lock-in*) | Medio | Medio | Moderado |
| |  | | --- | |  |   Integraciones incorrectas entre *React* con *TypeScript*, Django REST Framework y Cloud Run | Alta | Medio | Significativo |
| Brechas de seguridad o permisos mal configurados | Alta | Baja | Moderado |
| |  | | --- | |  |   Escalamiento futuro a *Vertex AI* no compatible o costoso | Medio | Medio | Moderado |
| |  | | --- | |  |   Falta de reentrenamientos periódicos del modelo IA | Alta | Medio | Significativo |

Tabla 13 Análisis cualitativo de la matriz de riesgos

|  |  |  |  |
| --- | --- | --- | --- |
| Criterio / Riesgo Analizado | Solución Propuesta | Alternativa CMMS Tradicional | Conclusión Comparativa |
| Adopción insuficiente del CMMS por parte de los usuarios en faena | Alta adopción por interfaz simple, similar a apps móviles. Riesgo medio. | Interfaces complejas, curva de aprendizaje alta. Riesgo alto. | Google reduce resistencia y facilita adopción en faena. |
| Calidad del *data entry* (inconsistencia, duplicados, registros incompletos) | Riesgo alto si no hay validaciones, pero *React* con *TypeScript* permite restricciones, listas, reglas y validadores. | Riesgo medio: algunos **CMMS** tienen validación, otros no. | Google permite más control directo y customizado por el equipo. |
| Fallas por modo offline o mala conectividad en faena | *Reac*t con *TypeScript* es fuerte en modo offline. Riesgo medio. | Varios **CMMS** dependen del módulo móvil premium y tienen offline limitado. | Google es más confiable para faena remota. |
| Limitaciones de *PostgreSQL* (*Cloud SQL*) / *Django REST Framework* (límites de ejecución o volumen) | Riesgo bajo–medio por límites, pero manejables con paginación o ***Cloud SQL*** opcional. | Riesgo bajo, infraestructura del proveedor. | **CMMS** es más estable, pero menos flexible. |
| Errores en predicciones del modelo IA (falsos positivos/negativos) | Riesgo alto si no hay reentrenamiento. Modelo ***Random Forest*** es interpretable. | Normalmente no incluye IA o solo módulos básicos. | Google permite IA propia y ajustable. |
| Dependencia del ecosistema Google (*lock-in*) | Riesgo bajo–medio (ecosistema Google). | Riesgo alto (dependencia total del proveedor, licencias caras). | **CMMS** tradicional tiene mayor *lock-in*. |
| Integraciones internas (*React* con *TypeScript* ↔ *Django REST Framework* ↔ *Cloud Run*) | Riesgo alto si no se gestionan bien los *endpoints* y *tokens*. | Riesgo medio (integraciones cerradas, pero más restrictivas). | Google permite más flexibilidad, pero requiere buena implementación. |
| Brechas de seguridad o permisos mal configurados | Riesgo medio (IAM + OIDC robusto, pero requiere configuración correcta). | Riesgo bajo (seguridad centralizada del proveedor). | Ambos son seguros, pero Google permite más control interno. |
| Escalamiento hacia *Vertex AI* no compatible o costoso | arquitectura ya compatible. | IA avanzada no suele estar incluida. | Google escala mejor a futuro. |
| Falta Reentrenamiento del modelo IA | Riesgo crítico si no se actualiza el *dataset.* | No aplica en la mayoría de **CMMS**. | Proyecto debe gestionar ciclo de vida del modelo. |

Tabla 14 Análisis Comparativo Cualitativo

### Análisis Cuantitativo

El análisis cuantitativo de los riesgos se lleva a cabo mediante los riesgos previamente identificados en el análisis cualitativo. Esto implica asignarles valores numéricos para poder tomar decisiones en situaciones de incertidumbre.

Tabla 15 Criterios del análisis cuantitativo

|  |  |  |
| --- | --- | --- |
| **Impacto** | | |
| =0 | $0 a $100.000 | >=$500.000 |
| **Probabilidad** | >=66% | Trivial | Tolerable | Moderado |
|  | 33% a 66% | Tolerable | Moderado | Significativo |
|  | <=33% | Moderado | Significativo | Critico |

|  |  |  |  |
| --- | --- | --- | --- |
| Riesgos Identificados | Impacto | Probabilidad | Prioridad |
| Adopción insuficiente del CMMS por parte de los usuarios en faena | >=$500.000 | 33% a 66% | Significativo |
| Baja calidad del *data entry* (inconsistencia, duplicados, registros incompletos) | >=$500.000 | 33% a 66% | Significativo |
| Fallas en aplicación web con capacidades offline o mala conectividad en faena | 33% a 66% | >=66% | Tolerable |
| Limitaciones de PostgreSQL (Cloud SQL) / Django REST Framework | 33% a 66% | 33% a 66% | Moderado |

|  |  |  |  |
| --- | --- | --- | --- |
| Errores en predicciones del modelo IA (falsos positivos/negativos) | >=$500.000 | >=$500.000 | Critico |
| Dependencia del ecosistema Google | 33% a 66% | 33% a 66% | Moderado |
| |  | | --- | |  |   Integraciones incorrectas entre *React* con *TypeScript*, *Django REST*, etc | >=$500.000 | 33% a 66% | Significativo |
| Brechas de seguridad o permisos mal configurados | >=$500.000 | >=66% | Moderado |
| |  | | --- | |  |   Escalamiento futuro a *Vertex AI* no compatible o costoso | 33% a 66% | 33% a 66% | Moderado |
| |  | | --- | |  |   Falta de reentrenamientos periódicos del modelo IA | >=$500.000 | 33% a 66% | Significativo |

Tabla 16 Análisis cuantitativo de la matriz de riesgos

|  |  |  |  |
| --- | --- | --- | --- |
| Riesgos Identificados | Solución Propuesta | CMMS Tradicional | Justificación Comparativa |
| Adopción insuficiente del CMMS por parte de los usuarios en faena | *React* con *TypeScript* es intuitivo, móvil y similar a usar WhatsApp o una app estándar; reduce resistencia al cambio. | Interfaces complejas, capacitación obligatoria y procesos rígidos suelen frenar uso real en faena. | *React* con *TypeScript* es intuitivo; CMMS suele tener curva de aprendizaje alta. |
| Baja calidad del *data entry* (inconsistencia, duplicados, registros incompletos) | *React* con *TypeScript* permite validaciones automáticas, listas controladas y restricciones, lo que disminuye errores. | Algunos CMMS tienen validaciones, pero suelen ser menos flexibles y dependen del proveedor. | Ambos dependen del usuario; *React* con *TypeScript* tiene mejores validadores. |
| Fallas en aplicación web con capacidades offline (PWA) o mala conectividad en faena | *React* con *TypeScript* almacena localmente datos y sincroniza al recuperar señal. | Requiere módulo móvil adicional y algunos sistemas no tienen offline real. | |  | | --- | |  |   *React* con *TypeScript* tiene offline nativo; algunos CMMS lo ofrecen con módulos pagados. |
| Limitaciones de *PostgreSQL* (*Cloud SQL*) / *Django REST Framework* (límites de ejecución o volumen) | Existen límites de filas, cuotas y tiempos de ejecución, aunque mitigables con *Cloud SQL* o *Cloud Run*. | CMMS usa infraestructura optimizada sin límites visibles para el usuario. | CMMS tiene infraestructura más robusta; Google tiene límites, pero se soluciones con *Cloud SQL/Run*. |

|  |  |  |  |
| --- | --- | --- | --- |
| Errores en predicciones del modelo IA (falsos positivos/negativos) | El uso de *Random Forest* mejora interpretabilidad, pero necesita reentrenamiento. | La mayoría de CMMS ni siquiera tiene IA, por lo que no existe riesgo de predicciones incorrectas. | La solución propuesta sí tiene IA real; CMMS no incluye IA. |
| Dependencia del ecosistema Google (*vendor lock-in*) | El sistema depende de *Google Cloud Platform* (GCP), aunque la arquitectura es portable. | **CMMS** tradicional bloquea datos, cobra por integración y dificulta migración. | Google tiene *lock-in* moderado, pero CMMS es mucho más restrictivo. |
| |  | | --- | |  |   Integraciones internas (*React* con *TypeScript* ↔ *Django REST Framework* ↔ *Cloud Run*) | Requiere buena configuración de *endpoints*, autenticación y manejo de *tokens*. | Es un sistema cerrado y rígido donde casi todo viene integrado de fábrica. | CMMS viene “cerrado”; Google requiere mayor integración técnica. |
| Brechas de seguridad o permisos mal configurados | IAM de Google es robusto, pero requiere configuraciones detalladas de roles. | Seguridad sólida controlada por el proveedor, pero menos flexible. | Ambos tienen seguridad sólida; Google requiere buena configuración. |
| |  | | --- | |  |   Escalamiento futuro a *Vertex AI* no compatible o costoso | El modelo actual en Cloud Run puede escalar sin reescritura. | La mayoría de CMMS no permite agregar IA avanzada. | Google escala nativamente; **CMMS** no contempla IA avanzada. |
| |  | | --- | |  |   Falta de reentrenamientos del modelo IA | El modelo necesita ciclos de actualización del *dataset* para mantener precisión. | No usan IA | CMMS no tiene IA → riesgo cero; tu solución sí debe gestionar ciclo de vida del modelo. |

Tabla 17 Análisis Comparativo Cualitativo

### Planes de respuesta

1.Adopción insuficiente del CMMS por parte de los usuarios en faena

Estrategia: Mitigar

Acciones:

* Implementar capacitación práctica y guiada dentro de la faena.
* Crear tutoriales simples (video corto + PDF).
* Incorporar ***feedback*** continuo mediante encuestas dentro de React con ***TypeScript***.
* Ajustar interfaz con “flujo por pasos” para reducir errores.
* Realizar pilotos controlados de 1–2 semanas antes del despliegue total.

2. Baja calidad de la ***data entry*** (inconsistencia, duplicados, registros incompletos)

Estrategia: Mitigar

Acciones:

* Implementar validadores obligatorios en React con ***TypeScript*** (listas, formatos, rangos).
* Configurar reglas automáticas en Django ***REST Framework*** para limpieza de datos.
* Crear ***Dashboards*** de monitoreo de calidad.
* Definir responsable de QA de datos en cada turno.
* Automatizar alertas cuando se detecten registros incompletos.

3. Fallas en aplicación ***web*** con capacidades ***offline*** (PWA) o conectividad limitada

Estrategia: Mitigar

Acciones:

* Configurar React con ***TypeScript*** en modo “***offline***-***first***” para formularios críticos.
* Reducir peso de imágenes mediante compresión automática.
* Crear protocolos de sincronización manual en zonas sin señal.
* Proveer puntos de Wi-Fi interno en zonas estratégicas (si la faena lo permite).

4. Limitaciones de PostgreSQL (***Cloud SQL***) / Django ***REST Framework*** (límites de ejecución o volumen)

Estrategia: Mitigar

Acciones:

* Implementar cache en ***Cloud*** ***Run*** y almacenamiento temporal en ***Cloud*** ***Storage***.
* Dividir planillas en subconjuntos (paginación) si superan 10k–20k filas.
* Migrar a ***Cloud SQL*** en caso de crecimiento elevado.
* Optimizar ***script***s reduciendo llamadas repetitivas a la API.

5. Errores en predicciones IA (falsos positivos o negativos)

Estrategia: Mitigar

Acciones:

* Programar reentrenamiento trimestral con datos actualizados.
* Validar los resultados con técnicos expertos antes de generar órdenes automáticas.
* Incluir un score de confianza para evitar acciones sobre predicciones débiles.
* Monitorear métricas del modelo (***accuracy***, ***recall***, ***precision***).

6. Dependencia del ecosistema Google (***vendor lock-in***)

Estrategia: Mitigar / Aceptar

Acciones:

* Mantener arquitectura basada en contenedores (***Cloud*** ***Run***) para facilitar portabilidad.
* Documentar API y flujos para una posible migración a AWS o Azure si fuese necesario.
* Mantener ***datasets*** exportables vía CSV/***Cloud*** ***Storage***.

7. Integraciones incorrectas entre React con ***TypeScript***, Django ***REST Framework*** y ***Cloud*** ***Run***

Estrategia: Mitigar

Acciones:

* Elaborar documentación técnica de cada ***endpoint***.
* Implementar pruebas unitarias y de integración.
* Configurar autenticación OAuth2 y ***tokens*** de servicio.
* Crear ***logs*** de auditoría en ***Cloud*** ***Logging***.

8. Brechas de seguridad o permisos mal configurados

Estrategia: Mitigar

Acciones:

* Aplicar IAM basado en mínimos privilegios.
* Revisiones mensuales de roles y accesos.
* Activar autenticación OIDC en ***Cloud*** ***Run***.
* Habilitar alertas de actividad sospechosa en ***Cloud*** ***Monitoring***.

9. Escalamiento futuro a Vertex AI no compatible o costoso

Estrategia: Mitigar

Acciones:

* Mantener el modelo almacenado en formato estándar (***joblib*** / ***pickle***).
* Documentar pipeline de entrenamiento y ***datasets***.
* Evaluar costos comparativos entre ***Cloud*** ***Run*** vs Vertex AI ***Workbench***.
* Diseñar arquitectura modular donde IA pueda reemplazarse sin afectar React con ***TypeScript***.

10. Falta de reentrenamientos del modelo IA

Estrategia: Mitigar

Acciones:

* Establecer calendario semestral o trimestral de reentrenamiento.
* Crear ***script*** automatizado para preparar ***dataset*** limpio desde el CMMS.
* Evaluar métricas tras cada reentrenamiento y registrar versiones del modelo.
* Crear ***Dashboard*** de salud del modelo (***drift***, precisión, casos especiales).

INGENIERÍA DEL SISTEMA

Tras haber establecido la viabilidad del proyecto y definido la metodología de trabajo, este capítulo profundiza en el diseño técnico detallado de la solución. La ingeniería del sistema constituye el puente entre los requerimientos de negocio abstractos y la implementación de *software* concreta, asegurando que cada componente cumpla una función específica dentro de la arquitectura general.

En las secciones siguientes se presenta la especificación exhaustiva de los requisitos funcionales y no funcionales, el modelado de datos que sustenta la persistencia de la información, y los diagramas de ingeniería (BPMN, Clases y Componentes) que documentan la lógica interna del sistema. Este nivel de detalle es fundamental para garantizar la trazabilidad, la mantenibilidad y la escalabilidad de la plataforma CMMS en el entorno productivo de la faena La Coipa.

## Especificación de Requisitos del Sistema

La especificación de requisitos traduce las necesidades del negocio en directrices técnicas precisas para el equipo de desarrollo. A continuación, se presenta el desglose detallado de los 30 Requerimientos Funcionales (RF) que componen la lógica del sistema y los Requerimientos No Funcionales (RNF) que establecen los estándares de calidad.

### Requisitos Funcionales (RF)

Los requisitos funcionales describen las interacciones específicas entre el usuario y el sistema, abarcando desde la gestión de usuarios hasta la inteligencia predictiva.

Tabla 18 Detalle de Requerimientos Funcionales

|  |  |  |
| --- | --- | --- |
| Código | Requerimiento | Descripción Detallada |
| RF-01 | Autenticación de Usuarios | El sistema debe permitir el inicio de sesión mediante credenciales (usuario/contraseña) utilizando tokens JWT seguros. Debe registrar intentos fallidos y permitir el cambio de contraseña. |
| RF-02 | Gestión de Usuarios y Roles | El administrador debe poder crear, editar y eliminar usuarios, asignando roles (Operador, Supervisor, Administrador) con permisos granulares específicos. |
| RF-03 | Gestión de Activos (Equipos) | Registro completo de la ficha técnica de cada activo (ID, marca, modelo, serie, estado operativo), permitiendo marcar equipos como "fuera de servicio”. |
| RF-04 | Visualización en Dashboard | Panel principal que muestra el estado de la flota en tiempo real mediante códigos de color, permitiendo filtrar por ubicación o tipo de equipo. |
| RF-05 | Creación de Órdenes de Trabajo (OT) | Funcionalidad para que los supervisores generen OTs (correctivas, preventivas, predictivas) con prioridad, descripción y fechas estimadas. |
| RF-06 | Asignación de OT | Capacidad para asignar una OT a un técnico específico, enviando una notificación automática al responsable vía Dashboard y Telegram. |
| RF-07 | Seguimiento de Estado de OT | El técnico debe poder actualizar el estado de la OT (En Progreso, Completada) y agregar comentarios técnicos sobre el hallazgo. |
| RF-08 | Registro de Tiempo | El sistema debe permitir imputar las horas-hombre dedicadas a cada OT, con capacidad de aprobación por parte del supervisor |
| RF-09 | Gestión de Repuestos en OT | Al ejecutar una OT, se debe poder especificar los repuestos utilizados, validando automáticamente su disponibilidad en el inventario. |
| RF-10 | Cierre y Finalización de OT | Flujo de aprobación donde el supervisor revisa la OT completada, valida los adjuntos (fotos/reportes) y cierra la orden definitivamente. |
| RF-11 | Creación de Checklists | Herramienta para diseñar plantillas de inspección personalizadas con múltiples ítems de verificación para los operadores. |
| RF-12 | Registro Digital de Checklists | Captura y almacenamiento de los *checklists* completados en terreno, vinculándolos al historial del equipo para alimentar el modelo de IA. |
| RF-13 | Generación de Alertas Predictivas | Ejecución periódica del modelo *Random Forest* para calcular la probabilidad de fallo. Si supera el umbral (ej. 85%), se genera una alerta automática. |
| RF-14 | Visualización de Alertas | Las alertas predictivas deben destacarse en el dashboard según su severidad, mostrando los factores causales (ej. T° alta) al hacer clic. |
| RF-15 | Creación Automática de OT por Alerta | Funcionalidad para convertir una alerta predictiva directamente en una OT pre-llenada, agilizando la respuesta ante riesgos inminentes. |
| RF-16 | Gestión de Inventario | Registro maestro de repuestos (código, stock, costo, ubicación) con capacidad para registrar entradas por compra y salidas por consumo. |
| RF-17 | Alertas de Inventario Bajo | Monitoreo automático del *stock*. Si un ítem cae bajo el mínimo crítico, el sistema debe alertar para gestionar la reposición. |
| RF-18 | Reportes de Inventario | Generación de informes de valoración de stock, rotación de repuestos e ítems obsoletos para optimizar la gestión de bodega. |
| RF-19 | Dashboard de KPIs | Visualización en tiempo real de indicadores clave: Disponibilidad Física (%), MTBF, MTTR y costos de mantenimiento acumulados. |
| RF-20 | Reportes de Mantenimiento | Generación de informes exportables (PDF/Excel) sobre OTs completadas, costos totales y equipos con mayor tasa de falla. |
| RF-21 | Historial de Vida del Equipo | Bitácora digital unificada que consolida todas las intervenciones, inspecciones y cambios de componentes de cada activo a lo largo del tiempo. |
| RF-22 | Notificaciones en Tiempo Real | Sistema de avisos visuales en el dashboard web ante eventos relevantes (nueva OT, alerta crítica), con marcado de lectura. |
| RF-23 | Bot Telegram: Notificaciones | Envío proactivo de mensajes *Push* al dispositivo del usuario ante alertas predictivas o asignaciones urgentes. |
| RF-24 | Bot Telegram: Consultas | Capacidad del bot para responder comandos de consulta (ej. /estado camioneta-01) entregando información resumida en el chat. |
| RF-25 | Exportación de Datos | Funcionalidad para descargar listados y reportes en formatos estándar (CSV/XLS) para análisis externo. |
| RF-26 | Auditoría y Trazabilidad | Registro inmutable (*Logs*) de todas las acciones críticas (creación, edición, borrado), indicando usuario y fecha para fines de auditoría. |
| RF-27 | Búsqueda Avanzada | Motor de búsqueda multicriterio para localizar rápidamente OTs, equipos o repuestos mediante filtros combinados. |
| RF-28 | Integración de Datos Externos | Capacidad para importar cargas masivas de datos históricos (ej. desde Excel) validando la integridad de la información. |
| RF-29 | Configuración del Sistema | Panel administrativo para ajustar parámetros globales: umbrales de alerta IA, prioridades de OT y niveles de stock mínimo. |
| RF-30 | Gestión de Permisos Granulares | Configuración fina de privilegios por rol, asegurando que cada perfil acceda estrictamente a las funciones necesarias para su labor. |

### Requisitos No Funcionales (RNF)

Los requisitos no funcionales definen los atributos de calidad y restricciones técnicas bajo las cuales debe operar el sistema para ser viable en el contexto minero.

Tabla 19 Detalle de Requerimientos No Funcionales

|  |  |  |
| --- | --- | --- |
| Código | Requerimiento | Criterio de Aceptación |
| RNF-01 | Seguridad | Implementación de JWT (Jones, Bradley, & Sakimura, 2015), encriptación HTTPS/TLS en tránsito y *hashing* de contraseñas (bcrypt). Auditoría de seguridad previa al despliegue. |
| RNF-02 | Rendimiento | Tiempo de carga del dashboard < 2 segundos. Capacidad para manejar >100 usuarios concurrentes. Ejecución del modelo IA < 5 minutos. |
| RNF-03 | Disponibilidad | *Uptime* garantizado del 99.5%. Plan de *Disaster Recovery* con *backups* automáticos diarios y recuperación sin pérdida de datos. |
| RNF-04 | Escalabilidad | Arquitectura capaz de escalar horizontalmente en la nube (PaaS) y base de datos optimizada para el crecimiento histórico de registros. |
| RNF-05 | Usabilidad | Interfaz intuitiva con curva de aprendizaje baja. Diseño *Mobile-First* optimizado para conexiones inestables en faena. |
| RNF-06 | Compatibilidad | Soporte total para navegadores modernos (Chrome, Firefox, Safari) y funcionamiento agnóstico en dispositivos móviles (Android/iOS). |
| RNF-07 | Mantenibilidad | Código documentado bajo estándares de la industria, incluyendo pruebas automatizadas (*Unit/Integration Tests*) para facilitar actualizaciones futuras. |
| RNF-08 | Precisión Predictiva | El modelo *Random Forest* debe mantener una métrica de Sensibilidad (*Recall*) superior al 90% para minimizar los falsos negativos en fallas críticas. |

## Modelado de Datos y Diagramas de Ingeniería

La arquitectura de la información y los flujos de procesos se diseñaron utilizando estándares de la industria (UML y BPMN) para asegurar una implementación coherente con la lógica de negocio de SOMACOR S.A. y garantizar la integridad referencial de los datos.

### Modelo de Datos Relacional (DER)

El esquema de persistencia se implementó sobre **PostgreSQL** (PostgreSQL Global Development Group, 2024), una base de datos relacional robusta seleccionada por su capacidad para manejar transacciones complejas y datos geoespaciales si fuera necesario en el futuro. El diseño se estructura en torno a tres entidades principales que soportan la lógica del negocio:

* **Activos (Asset):** Entidad maestra que actúa como la "fuente de verdad". Almacena la ficha técnica, ubicación, estado operativo actual y metadatos críticos para el modelo predictivo (marca, modelo, año).
* **Órdenes de Trabajo (WorkOrder):** Entidad transaccional que vincula un Activo con un Usuario responsable (assigned\_to). Registra el ciclo de vida del mantenimiento, incluyendo atributos de control como prioridad, estado (status), fecha programada y cierre técnico.
* **Predicciones de Fallo (FailurePrediction):** Entidad analítica generada automáticamente por el motor de IA. Almacena el historial de probabilidades calculadas (failure\_probability) y el nivel de riesgo clasificado (risk\_level), permitiendo auditorías sobre la precisión del modelo a lo largo del tiempo.

Diagrama, Esquemático

El contenido generado por IA puede ser incorrecto.

Figura 10 Diagrama Entidad-Relación – DER

**1. Gestión de Activos y Ubicaciones**

Este módulo define la "fuente de verdad" sobre la flota y su localización física.

* **Activo (Activo):** Entidad maestra que almacena la ficha técnica del equipo. Contiene atributos críticos para la identificación (Codigo\_Activo, Matricula, Numero\_Serie) y variables que alimentan el modelo predictivo, como Fabricante, Modelo y Fecha\_De\_Instalacion. Además, gestiona el estado operativo (Criticidad, Estado).
* **Ubicación (Ubicacion):** Normaliza los lugares físicos de la faena (ej. "Taller Central", "Frente Norte"), permitiendo agrupar los activos geográficamente.

**2. Gestión del Mantenimiento y Órdenes de Trabajo**

El núcleo transaccional del sistema, encargado de orquestar la ejecución de tareas.

* **Orden de Trabajo (Orden\_Trabajo):** Entidad central que registra el ciclo de vida del mantenimiento. Vincula un **Activo** con un **Usuario** responsable. Controla el flujo mediante estados (Pendiente, En Progreso, Completado\_En) y registra métricas de desempeño como Horas\_Estimadas vs Horas\_Reales.
* **Plan de Mantenimiento (Plan\_Mantenimiento):** Define las estrategias recurrentes (Preventivas). Establece la Frecuencia y el Tipo\_Recurrencia para la generación automática de OTs.
* **Listas de Verificación (Plantilla\_Lista y Lista\_Verificacion\_Repuestos):** Implementa la digitalización de los *checklists*.
  + Plantilla\_Lista: Define la estructura del formulario (qué se debe revisar).
  + Lista\_Verificacion\_Repuestos: Es la instancia ejecutada, que almacena las respuestas, la puntuación obtenida, la aprobación y el enlace al PDF generado (Url\_Pdf).

**3. Inteligencia Artificial y Predicciones**

Módulo analítico diseñado para almacenar los resultados del motor de Machine Learning.

* **Predicción de Fallos (Prediccion\_Fallos):** Tabla histórica que almacena la salida del algoritmo *Random Forest*. Registra la Probabilidad\_Fallo, el Nivel\_Riesgo, la Fecha\_Fallo\_Predicha y, crucialmente, la Caracteristica\_Encontrada (explicabilidad del modelo) y la Version\_Modelo utilizada.
* **Alerta (Alerta):** Gestiona los avisos críticos generados por el sistema (ej. "Riesgo Alto Detectado"), controlando su estado de lectura y resolución.

**4. Gestión de Inventario y Repuestos**

Controla el consumo de materiales asociado a las intervenciones.

* **Pieza de Repuesto (Pieza\_Repuesto):** Maestro de materiales. Controla el Stock\_Minimo, Costo\_Unitario y la ubicación en bodega.
* **Movimiento de Stock (Movimiento\_Stock):** Registro de auditoría (Kardex) que traza cada entrada o salida de material, vinculando el consumo directamente a una Orden\_Trabajo para el cálculo de costos reales.

**5. Usuarios y Seguridad**

Administra el acceso y la comunicación omnicanal.

* **Usuario (Usuario):** Almacena las credenciales, datos personales y el ID\_Telegram necesario para el envío de notificaciones *push*.
* **Rol (Rol):** Define los perfiles de acceso (Administrador, Supervisor, Técnico, Operador).
* **Notificación (Notificacion):** Log de comunicaciones enviadas al usuario, indicando si fueron leídas y si se transmitieron vía Telegram.

### Diccionario de Datos

A continuación, se detallan las tablas junto con sus campos y atributos además de incorporar una descripción sobre el campo:

**1. Gestión de Activos y Ubicaciones**

Tabla 20 Ubicación

|  |  |  |  |
| --- | --- | --- | --- |
| Atributo | Tipo Sugerido | Descripción | Llave |
| ID\_Ubicacion | INT / UUID | Identificador único de la ubicación. | PK |
| Nombre | VARCHAR | Nombre de la ubicación. |  |
| Codigo | VARCHAR | Código corto para identificar la ubicación. |  |
| Descripcion | TEXT | Detalles adicionales del lugar. |  |
| Creado\_En | DATETIME | Fecha de creación del registro. |  |

Tabla 21 Activo

|  |  |  |  |
| --- | --- | --- | --- |
| Atributo | Tipo Sugerido | Descripción | Llave |
| ID\_Activo | INT / UUID | Identificador único del activo. | PK |
| Nombre | VARCHAR | Nombre común del activo. |  |
| Codigo\_Activo | VARCHAR | Código interno de inventario. |  |
| Tipo\_Vehiculo | VARCHAR | Clasificación del vehículo o máquina. |  |
| Fabricante | VARCHAR | Marca o fabricante. |  |
| Modelo | VARCHAR | Modelo del activo. |  |
| Numero\_Serie | VARCHAR | Número de serie. |  |
| Matricula | VARCHAR | Patente o matrícula. |  |
| Fecha\_De\_Instalacion | DATE | Fecha de puesta en marcha. |  |
| Estado | VARCHAR | Estado actual del activo. |  |
| Criticidad | VARCHAR | Nivel de importancia. |  |
| Especificaciones | JSON / TEXT | Detalles técnicos adicionales. |  |
| Creado\_En | DATETIME | Fecha de registro. |  |
| Actualizado\_En | DATETIME | Última modificación. |  |
| ID\_Ubicacion | INT / UUID | Referencia a la ubicación. | FK |

**2. Mantenimiento y Órdenes de Trabajo**

Tabla 22 Plan\_Mantenimiento

|  |  |  |  |
| --- | --- | --- | --- |
| Atributo | Tipo Sugerido | Descripción | Llave |
| ID\_PMantenimiento | INT / UUID | Identificador del plan. | PK |
| Nombre | VARCHAR | Nombre del plan de mantenimiento. |  |
| Tipo\_Plan | VARCHAR | Preventivo, Predictivo, etc. |  |
| Tipo\_Recurrencia | VARCHAR | Basado en tiempo, uso, etc. |  |
| Intervalo\_Recurrencia | INT | Valor numérico del intervalo. |  |
| Fecha\_Vencimiento | DATE | Fecha límite para la ejecución. |  |
| Estado\_Activo | BOOLEAN | Indica si el plan está vigente. |  |
| Duracion\_Estimada | DECIMAL | Tiempo estimado (horas). |  |
| Creado\_Por | VARCHAR | Usuario que creó el plan. |  |
| ID\_Plantilla\_Lista | INT / UUID | Checklist asociado. | FK |
| ID\_Activo | INT / UUID | Activo al que aplica. | FK |

Tabla 23 Orden\_Trabajo

|  |  |  |  |
| --- | --- | --- | --- |
| Atributo | Tipo Sugerido | Descripción | Llave |
| ID\_Orden\_Trabajo | INT / UUID | Identificador de la OT. | PK |
| Numero\_OT | VARCHAR | Número legible de OT. |  |
| Titulo | VARCHAR | Título del trabajo. |  |
| Descripcion | TEXT | Descripción de la tarea. |  |
| Tipo\_OT | VARCHAR | Correctiva, Preventiva, etc. |  |
| Prioridad | VARCHAR | Alta, Media, Baja. |  |
| Estado | VARCHAR | Pendiente, En Progreso, Completada. |  |
| Fecha\_Programada | DATETIME | Fecha planificada. |  |
| Iniciado\_En | DATETIME | Fecha real de inicio. |  |
| Completado\_En | DATETIME | Fecha real de fin. |  |
| Horas\_Estimadas | DECIMAL | Tiempo planificado. |  |
| Horas\_Reales | DECIMAL | Tiempo real usado. |  |
| Nota\_Finalizacion | TEXT | Comentarios finales. |  |
| ID\_Usuario | INT / UUID | Técnico asignado. | FK |

Tabla 24 Plantilla\_Lista

|  |  |  |  |
| --- | --- | --- | --- |
| Atributo | Tipo Sugerido | Descripción | Llave |
| ID\_Plantilla\_Lista | INT / UUID | Identificador del checklist. | PK |
| Nombre | VARCHAR | Nombre de la plantilla. |  |
| Codigo | VARCHAR | Código de referencia. |  |
| Tipo\_Vehiculo | VARCHAR | Tipo de activo aplicable. |  |
| Descripcion | TEXT | Instrucciones generales. |  |
| Elemento | JSON / TEXT | Ítems del checklist. |  |
| Es\_Plantilla\_Sistema | BOOLEAN | Si es una plantilla por defecto. |  |
| Puntuacion\_Aprobado | DECIMAL | Puntaje mínimo. |  |

Tabla 25 Lista\_Verificacion\_Repuestos

|  |  |  |  |
| --- | --- | --- | --- |
| Atributo | Tipo Sugerido | Descripción | Llave |
| ID\_Lista\_Verificacion | INT / UUID | Identificador del checklist ejecutado. | PK |
| Repuestos | JSON / TEXT | Repuestos utilizados. |  |
| Puntuacion | DECIMAL | Calificación del checklist. |  |
| Aprobado | BOOLEAN | Resultado de evaluación. |  |
| Url\_Pdf | VARCHAR | Enlace al PDF generado. |  |
| Completado\_En | DATETIME | Fecha de cierre. |  |
| ID\_Orden\_Trabajo | INT / UUID | OT asociada. | FK |
| ID\_Plantilla\_Lista | INT / UUID | Plantilla usada. | FK |

**3. Inventario y Repuestos**

Tabla 26 Pieza\_Repuesto

|  |  |  |  |
| --- | --- | --- | --- |
| Atributo | Tipo Sugerido | Descripción | Llave |
| ID\_Pieza\_Repuesto | INT / UUID | Identificador del repuesto. | PK |
| Numero\_Pieza | VARCHAR | SKU o código de pieza. |  |
| Nombre | VARCHAR | Nombre del repuesto. |  |
| Descripcion | TEXT | Información técnica. |  |
| Categoria | VARCHAR | Eléctrico, Mecánico, etc. |  |
| Cantidad | INT | Stock actual. |  |
| Stock\_Minimo | INT | Umbral mínimo. |  |
| Costo\_Unitario | DECIMAL | Precio unitario. |  |
| Ubicacion | VARCHAR | Ubicación física. |  |
| Proveedor | VARCHAR | Proveedor asociado. |  |

Tabla 27 Movimiento\_Stock

|  |  |  |  |
| --- | --- | --- | --- |
| Atributo | Tipo Sugerido | Descripción | Llave |
| ID\_Movimiento\_Stock | INT / UUID | Identificador del movimiento. | PK |
| Tipo\_Movimiento | VARCHAR | Entrada o salida. |  |
| Cantidad | INT | Unidades movidas. |  |
| Notas | TEXT | Justificación del movimiento. |  |
| ID\_Pieza\_Repuesto | INT / UUID | Repuesto afectado. | FK |
| ID\_Orden\_Trabajo | INT / UUID | OT relacionada (si aplica). | FK |

**4. Monitoreo y Predicción**

Tabla 28 Prediccion\_Fallos

|  |  |  |  |
| --- | --- | --- | --- |
| Atributo | Tipo Sugerido | Descripción | Llave |
| ID\_Prediccion\_Fallos | INT / UUID | Identificador de predicción. | PK |
| Fecha\_Prediccion | DATETIME | Fecha del análisis. |  |
| Probabilidad\_Fallo | DECIMAL | 0.0 a 1.0. |  |
| Puntuacion\_Confianza | DECIMAL | Nivel de certeza. |  |
| Fecha\_Fallo\_Predicha | DATE | Fecha estimada del fallo. |  |
| Caracteristica\_Encontrada | VARCHAR | Anomalía detectada. |  |
| Version\_Modelo | VARCHAR | Versión del modelo IA. |  |
| Recomendacion | TEXT | Acción sugerida. |  |
| Nivel\_Riesgo | VARCHAR | Alto, Medio, Bajo. |  |
| ID\_Activo | INT / UUID | Activo analizado. | FK |

Tabla 29 Alerta

|  |  |  |  |
| --- | --- | --- | --- |
| Atributo | Tipo Sugerido | Descripción | Llave |
| ID\_Tipo\_Alerta | INT / UUID | Identificador de la alerta. | PK |
| Tipo\_Alerta | VARCHAR | Categoría. |  |
| Gravedad | VARCHAR | Severidad. |  |
| Titulo | VARCHAR | Resumen. |  |
| Mensaje | TEXT | Descripción completa. |  |
| Leido | BOOLEAN | Marcado como leído. |  |
| Resuelto | BOOLEAN | Marcado como resuelto. |  |
| ID\_Activo | INT / UUID | Activo que generó la alerta. | FK |

**5. Usuarios y Seguridad**

Tabla 30 Usuario

|  |  |  |  |
| --- | --- | --- | --- |
| Atributo | Tipo Sugerido | Descripción | Llave |
| ID\_Usuario | INT / UUID | Identificador del usuario. | PK |
| Correo | VARCHAR | Email (login). |  |
| Nombre\_Apellido | VARCHAR | Nombre y apellido. |  |
| Contrasenia | VARCHAR | Hash de contraseña. |  |
| Rut | VARCHAR | Identificación nacional. |  |
| ID\_Telegram | VARCHAR | ID para alertas. |  |
| Tipo\_Licencia | VARCHAR | Tipo de licencia. |  |
| Caducidad\_Licencia | DATE | Vencimiento. |  |
| ID\_Rol | INT / UUID | Rol asignado. | FK |

Tabla 31 Rol

|  |  |  |  |
| --- | --- | --- | --- |
| Atributo | Tipo Sugerido | Descripción | Llave |
| ID\_Rol | INT / UUID | Identificador del rol. | PK |
| Nombre | VARCHAR | Nombre del rol. |  |
| Descripcion | TEXT | Permisos del rol. |  |

Tabla 32 Notificacion

|  |  |  |  |
| --- | --- | --- | --- |
| Atributo | Tipo Sugerido | Descripción | Llave |
| ID\_Notificacion | INT / UUID | Identificador de notificación. | PK |
| Tipo\_Notificacion | VARCHAR | Email, Push, Telegram. |  |
| Titulo\_Mensaje | TEXT | Contenido del mensaje. |  |
| Datos | JSON | Datos técnicos adjuntos. |  |
| Enviado\_Por\_Telegram | BOOLEAN | Flag de envío externo. |  |
| ID\_Usuario | INT / UUID | Destinatario. | FK |

### Modelado de Procesos de Negocio (BPMN To-Be)

Para operacionalizar la estrategia predictiva, se diseñó un nuevo flujo de trabajo que integra la inteligencia artificial con la toma de decisiones humana. A diferencia del proceso manual anterior, este flujo "To-Be" es proactivo y automatizado.

El ciclo de **Mantenimiento Predictivo** se orquesta de la siguiente manera:

1. **Ingesta y Análisis:** El sistema ejecuta diariamente una tarea programada (*Celery Beat*) que alimenta el modelo *Random Forest* con los datos operativos más recientes.
2. **Detección de Anomalía:** Si el algoritmo estima una probabilidad de fallo superior al umbral crítico (ej. >85%), se instancia automáticamente una alerta de riesgo.
3. **Notificación Omnicanal:** El sistema dispara una alerta en tiempo real al Supervisor a través del Bot de Telegram, eliminando la latencia del correo electrónico o el papel.
4. **Toma de Decisión y Ejecución:** El Supervisor valida la alerta desde su dispositivo móvil. Si es positiva, el sistema genera automáticamente una Orden de Trabajo (OT) Predictiva para que el técnico ejecute la intervención antes de que ocurra la falla.

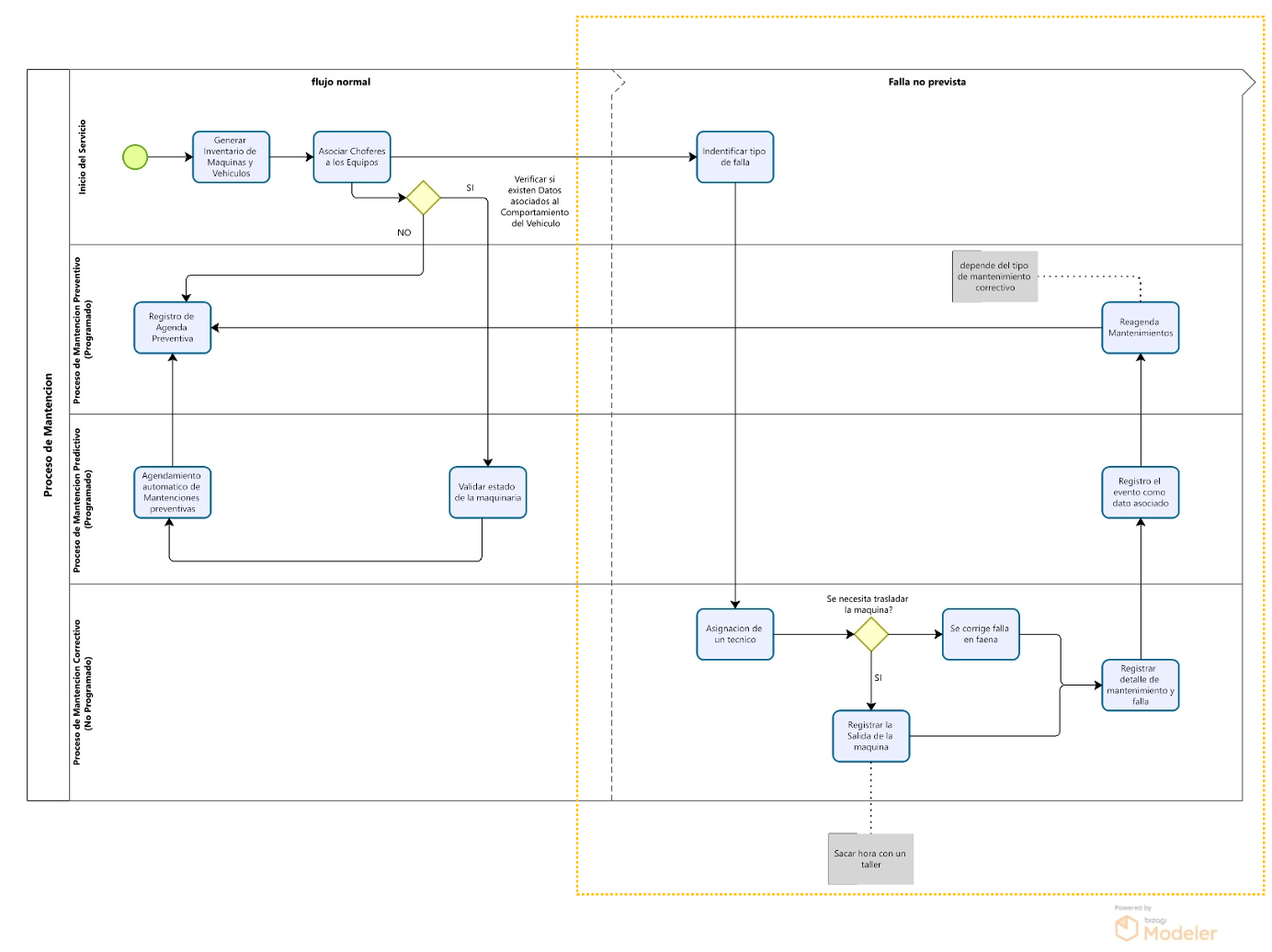


Figura 11 Diagrama BPMN del Ciclo de Mantenimiento Predictivo Propuesto (To-Be).

### Diagrama de Clases y Lógica de Negocio

La implementación en el *backend* (Django) sigue un patrón orientado a objetos que facilita la extensibilidad y el mantenimiento del código. Las clases principales están diseñadas para desacoplar la lógica de presentación de la lógica de negocio:

* **FailurePredictionTrainer:** Clase que encapsula toda la lógica de ciencia de datos. Contiene métodos para el pre-procesamiento de *datasets*, el entrenamiento del modelo *Random Forest* y la serialización del objeto entrenado para su uso en producción.
* **BotCommandHandler:** Clase controladora que centraliza la lógica de interacción del usuario en Telegram. Interpreta los comandos entrantes (ej. /status) y despacha las respuestas adecuadas, manteniendo la interfaz de chat separada de las reglas de negocio del núcleo.

Diagrama, Dibujo de ingeniería

El contenido generado por IA puede ser incorrecto.

Figura 12 Diagrama de Clases UML

## Diseño de la Arquitectura del *Software*

Para garantizar la escalabilidad, mantenibilidad y la alta disponibilidad requerida por una operación minera de flujo continuo, se diseñó una **Arquitectura de Microservicios Desacoplada** (*Headless Architecture*). Este diseño separa estrictamente la capa de procesamiento y lógica de negocio (*Backend*) de la capa de presentación (*Frontend*), permitiendo que ambos componentes evolucionen y escalen de manera independiente.

### Diagrama de Infraestructura y Despliegue

La infraestructura se orquestó bajo el paradigma de "Plataforma como Servicio" (PaaS) en la nube, priorizando servicios gestionados para reducir la carga administrativa de mantenimiento de servidores. El despliegue se realiza en un entorno contenerizado (Merkel, 2014), lo que asegura la consistencia entre los ambientes de desarrollo y producción.

Imagen que contiene Diagrama

El contenido generado por IA puede ser incorrecto.

Figura 13 Diagrama de Arquitectura de Despliegue

### Descripción de Componentes

El diagrama ilustra la segregación de responsabilidades en cuatro capas críticas para el rendimiento del sistema:

1. **Capa de Presentación (*Frontend - Vercel*):** La interfaz de usuario, desarrollada como una *Single Page Application* (SPA) en React, se despliega en la red de entrega de contenido (CDN) global de **Vercel**. Esto garantiza una baja latencia en la carga de la interfaz para los operadores, independientemente de su ubicación geográfica, y desacopla la disponibilidad visual del estado del servidor de procesamiento.
2. **Capa de Procesamiento (*Backend - Railway*):** La lógica de negocio reside en **Railway**, subdividida en servicios independientes para evitar puntos únicos de fallo:
   * **Servicio Web (Django/Gunicorn):** Maneja exclusivamente las solicitudes HTTP de la API REST y los *webhooks* de Telegram. Su función es responder rápidamente al usuario.
   * **Servicio Worker (Celery):** Un proceso aislado dedicado a la ejecución de tareas pesadas en segundo plano, específicamente el entrenamiento del modelo *Random Forest* y la generación masiva de predicciones. Esta separación es vital para impedir que la carga computacional de la IA degrade el rendimiento de la plataforma web.
3. **Capa de Intermediación (Broker - Redis):** Se implementa **Redis** (Redis Ltd, 2024) como un bus de mensajes en memoria de alto rendimiento. Actúa como el intermediario que comunica al servicio Web con el *Worker*, gestionando las colas de tareas de manera asíncrona y asegurando que ninguna solicitud de procesamiento se pierda en caso de saturación momentánea.
4. **Capa de Persistencia (Base de Datos):** Los datos maestros (activos, usuarios, bitácoras) se almacenan en una base de datos relacional **PostgreSQL** gestionada. Esta instancia cuenta con copias de seguridad automáticas y aislamiento de red, cumpliendo con los estándares de integridad y seguridad de la información del SGI[[1]](#footnote-1).

### Flujo de Comunicación Omnicanal

La arquitectura integra externamente la API de **Telegram** mediante *Webhooks*. Cuando el *Worker* de Celery detecta una anomalía crítica tras procesar el modelo de IA, inyecta un mensaje en la cola de salida hacia la API de Telegram, la cual se encarga de realizar el *Push* de la notificación al dispositivo del supervisor, cerrando el ciclo de comunicación en tiempo real.

## Diseño de Interfaz de Usuario (UI/UX)

El diseño de la interfaz se desarrolló bajo el principio de *Mobile-First*, priorizando la claridad visual y la eficiencia en la interacción para reducir la carga cognitiva de los operadores y supervisores en un entorno de alta exigencia como la faena La Coipa. La capa de presentación, construida en React[[2]](#footnote-2), implementa componentes responsivos que se adaptan a estaciones de trabajo de escritorio y dispositivos móviles.

### Tablero de Control Operacional (*Dashboard*)

Esta vista actúa como el centro de mando estratégico. Su diseño implementa el patrón de "Semáforo Visual", permitiendo a la gerencia evaluar el estado de la flota en segundos.

* **Panel de KPIs:** Visualización de indicadores críticos (Disponibilidad, MTBF) mediante gráficos de velocímetro y barras, facilitando la lectura rápida.
* **Mapa de Estado de Flota:** Lista priorizada de equipos, donde los activos con alertas predictivas activas ("Riesgo de Falla") se destacan automáticamente en la parte superior con distintivos rojos, seguidos por equipos en mantenimiento (amarillo) y operativos (verde).

Captura de pantalla de computadora

El contenido generado por IA puede ser incorrecto.

Figura 14 Captura de pantalla del Dashboard Principal

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

Figura 15 Captura de pantalla del Dashboard Principal parte 2

Una captura de pantalla de una red social

El contenido generado por IA puede ser incorrecto.

Figura 16 Captura de pantalla del Dashboard Principal parte 3

### Gestión de Activos y Hoja de Vida Digital

La interfaz de gestión de activos reemplaza los archivadores físicos. Se diseñó una vista de "Detalle de Equipo" que centraliza toda la información técnica y el historial de intervenciones.

* **Ficha Técnica:** Datos estáticos del equipo (Modelo, Serie, Año) editables solo por administradores.
* **Línea de Tiempo (Timeline):** Visualización cronológica de todas las Órdenes de Trabajo y alertas predictivas asociadas al activo, garantizando la trazabilidad completa exigida por la norma ISO 9001.

Interfaz de usuario gráfica, Aplicación, Teams

El contenido generado por IA puede ser incorrecto.

Figura 17 Captura de pantalla del Perfil de un Activo/Equipo

### Flujo de Órdenes de Trabajo (OT)

La interfaz para las OTs[[3]](#footnote-3) se diseñó pensando en la operatividad en terreno. Los formularios de ingreso son simplificados, utilizando selectores desplegables y casillas de verificación para minimizar la escritura manual.

* **Tablero Kanban:** Visualización del flujo de trabajo (Pendiente →to→ En Progreso →to→ Terminada) que permite a los supervisores arrastrar y soltar tareas para cambiar su estado o asignación.
* **Formulario de Cierre:** Interfaz específica para el técnico que permite registrar horas, repuestos consumidos y adjuntar evidencia fotográfica desde el dispositivo móvil.

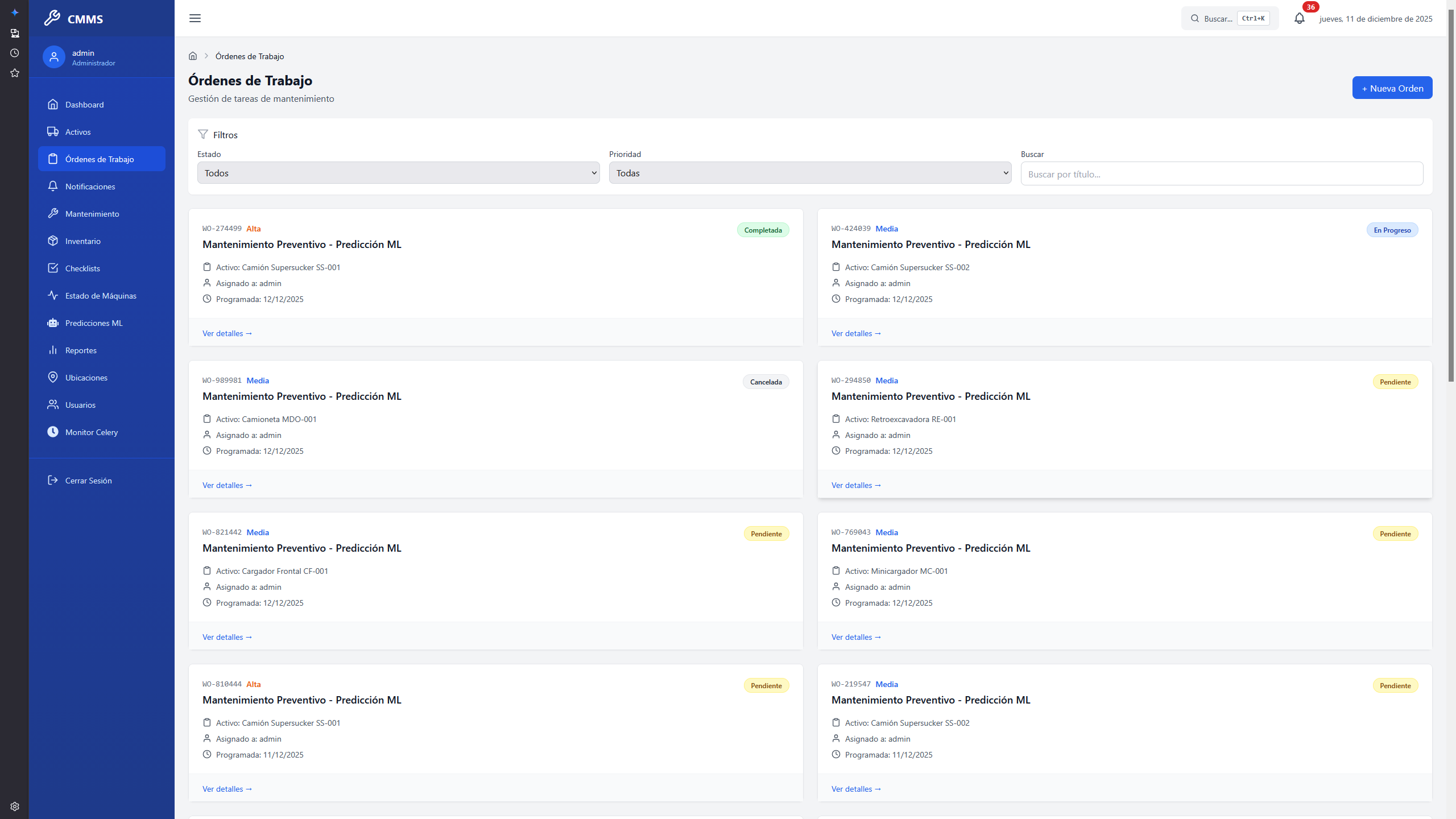


Figura 18 Captura de pantalla de la Gestión de OTs

### Interfaz Omnicanal (Telegram Bot)

A diferencia de las pantallas web, la interfaz del Bot es conversacional y minimalista, diseñada para funcionar con un consumo de datos mínimo.

* **Menú de Comandos:** Acceso rápido a funciones frecuentes mediante un botón de menú persistente (/status, /mis\_ordenes).
* **Teclados Interactivos (*Inline Keyboards*):** Botones integrados en las respuestas del chat (ej. "Aceptar OT", "Ver Detalle") que permiten interactuar con el sistema sin necesidad de escribir texto, reduciendo errores y agilizando la respuesta.

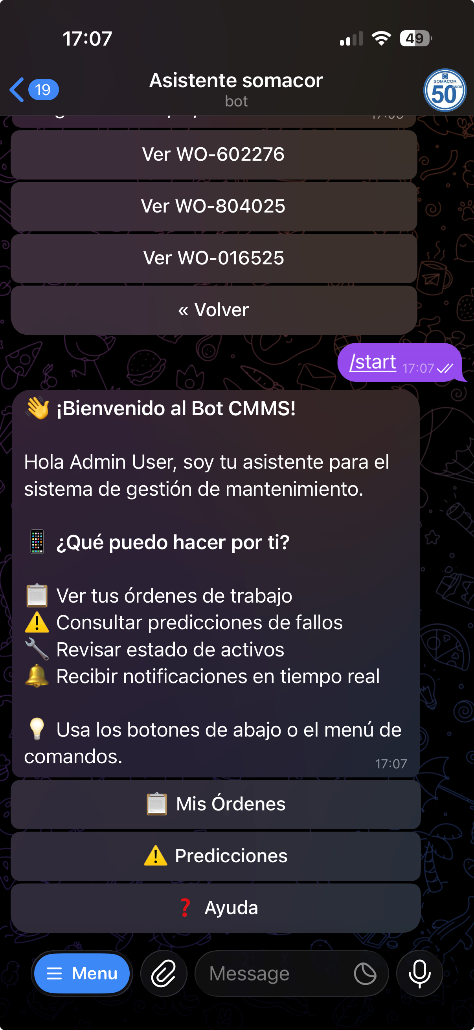
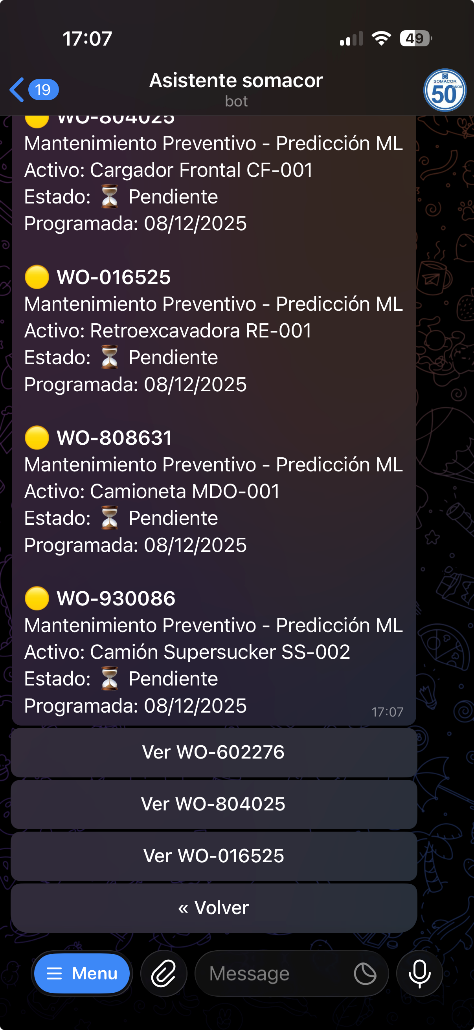


Figura 19 Captura de pantalla de la interacción con el Bot en Telegram parte 2

Figura 20 Captura de pantalla de la interacción con el Bot en Telegram

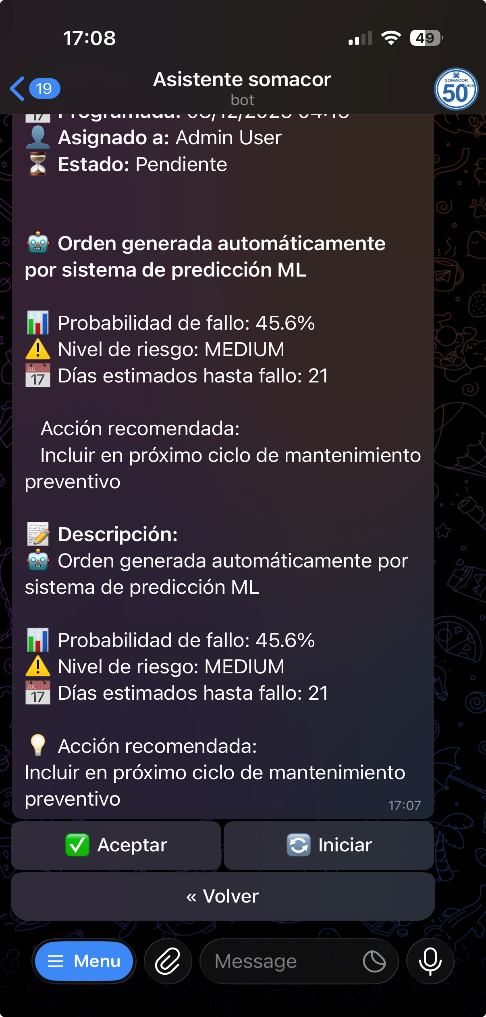


Figura 21 Captura de pantalla de la interacción con el Bot en Telegram parte 3

CONCLUSIONES Y RECOMENDACIONES

La culminación del proyecto "Implementación de un Sistema de Gestión de Mantenimiento Computarizado (CMMS) Predictivo" representa un hito significativo en la modernización de los procesos operativos de SOMACOR S.A. en la faena La Coipa. Este trabajo ha demostrado que la integración de tecnologías de la Industria 4.0, cuando se aplica con un enfoque de ingeniería centrado en el usuario, es capaz de resolver problemáticas estructurales de eficiencia y seguridad.

A continuación, se presentan las conclusiones derivadas del desarrollo del sistema, así como las recomendaciones estratégicas para garantizar su sostenibilidad y evolución en el tiempo.

## Conclusiones

Los resultados obtenidos permiten validar la hipótesis planteada al inicio de esta investigación, confirmando que la implementación de una plataforma digital con capacidades predictivas reduce la incertidumbre operacional y optimiza la gestión de activos críticos.

1. **Validación de la Estrategia "Cold Start":** Uno de los hallazgos técnicos más relevantes fue la eficacia de la estrategia de generación de datos sintéticos basados en conocimiento experto. Se demostró que es posible desplegar un modelo de Inteligencia Artificial funcional (*Random Forest*) sin depender de un histórico de datos previo, alcanzando una métrica de Sensibilidad (*Recall*) superior al 85% en las pruebas de validación. Esto rompe la barrera de entrada tradicional para la adopción de IA en faenas con baja madurez digital.
2. **Transformación del Modelo de Gestión:** El sistema logró erradicar la "ceguera de datos" diagnosticada inicialmente. La transición desde bitácoras en papel hacia un registro digital centralizado ha dotado a la gerencia de una trazabilidad completa sobre el ciclo de vida de los equipos, permitiendo transitar desde un enfoque reactivo (*Run-to-Failure*) hacia una cultura proactiva basada en la evidencia.
3. **Reducción de la Fricción Tecnológica:** La integración del Bot Omnicanal en Telegram probó ser una decisión de diseño acertada para el contexto de la faena. Al llevar la interfaz de gestión al dispositivo móvil personal del operador, se redujo significativamente la latencia en el reporte de hallazgos y se facilitó la adopción del sistema por parte de usuarios no nativos digitales, minimizando la resistencia al cambio.
4. **Robustez de la Arquitectura Desacoplada:** La arquitectura *Headless* (Django + React) demostró ser resiliente y escalable. La separación de responsabilidades permitió que el entrenamiento intensivo de los modelos predictivos no degradara el rendimiento de la interfaz de usuario, garantizando la continuidad del servicio incluso bajo condiciones de conectividad inestable.

## Recomendaciones para la Organización

Para maximizar el retorno de la inversión (ROI) y asegurar la sostenibilidad del sistema, se sugieren las siguientes acciones a nivel organizacional:

* **Institucionalizar el Re-entrenamiento:** Se recomienda establecer un protocolo mensual para re-entrenar el modelo predictivo con los nuevos datos reales recolectados en faena. Esto permitirá que el algoritmo evolucione y se adapte a las condiciones cambiantes de la operación (ej. envejecimiento de la flota), mitigando el riesgo de degradación del modelo (*Model Drift*).
* **Plan de Incentivos a la Digitalización:** Para consolidar el uso de la herramienta, se sugiere vincular indicadores de desempeño individual (como la cantidad de *checklists* digitales completados a tiempo) con el sistema de incentivos de los operadores, reforzando la cultura de registro digital.
* **Auditoría de Calidad del Dato:** Implementar revisiones trimestrales de la calidad de la información ingresada por los técnicos para detectar vicios de operación o necesidades de re-capacitación en el uso de los formularios móviles.

## Lecciones Aprendidas

El desarrollo e implementación del sistema CMMS en un entorno minero real ha proporcionado valiosas enseñanzas que trascienden lo puramente técnico. Estas lecciones se documentan a continuación como base de conocimiento para futuros proyectos de innovación tecnológica en la organización.

1. **La Experiencia de Usuario (UX) define la Adopción:** Se evidenció que la resistencia al cambio no se debe necesariamente a una negativa a la tecnología, sino a la complejidad percibida de las herramientas. La decisión de integrar un **Bot de Telegram** como interfaz simplificada demostró ser crítica: al utilizar un canal de comunicación familiar para los operadores, se redujo la barrera de entrada y se aceleró la adopción del sistema mucho más rápido que si se hubiera impuesto una aplicación web tradicional compleja.
2. **Calidad de Datos sobre Complejidad del Algoritmo:** Durante el entrenamiento del modelo predictivo, se comprobó que la precisión del algoritmo *Random Forest* depende más de la calidad y limpieza de los datos de entrada (códigos de falla normalizados) que de la sofisticación del modelo mismo. La estrategia de validar los datos en el punto de ingreso (formularios con campos restringidos) resultó ser más efectiva para mejorar las predicciones que cualquier ajuste técnico de hiperparámetros.
3. **Arquitectura "Headless" como Seguro de Vida:** La elección de desacoplar el *Frontend* (React) del *Backend* (Django) pagó dividendos inmediatos. Esta separación permitió realizar mantenimientos correctivos y ajustes en la lógica de negocio del servidor sin interrumpir la disponibilidad visual de la aplicación para los usuarios en terreno, garantizando una continuidad operativa vital para la faena.

## Trabajos Futuros

Dada la naturaleza escalable de la solución, se identifican oportunidades técnicas para expandir el alcance del proyecto en futuras iteraciones:

* **Integración IoT (*Internet of Things*):** La evolución natural del sistema es la incorporación de dispositivos telemáticos (GPS/CanBus) en la maquinaria para la captura automática de horómetros y códigos de falla. Esto eliminaría la dependencia del ingreso manual de datos, aumentando la precisión del modelo predictivo al 100% de confiabilidad y permitiendo un monitoreo en tiempo real.
* **Módulo de Abastecimiento Predictivo:** Desarrollar una integración directa con el sistema de adquisiciones (ERP) para automatizar la compra de repuestos. Al cruzar la predicción de fallo con el nivel de *stock*, el sistema podría generar órdenes de compra automáticas bajo la metodología "Justo a Tiempo" (*Just-in-Time*), optimizando el flujo de caja de la empresa.
* **Expansión a Nuevas Variables:** Incorporar variables exógenas al modelo, como los pronósticos meteorológicos de la zona (invierno altiplánico), para refinar la precisión de las alertas en periodos de condiciones climáticas extremas que afectan la viscosidad de lubricantes y el rendimiento de motores.
* **Evolución hacia MLOps y Cloud Enterprise (GCP/Vertex AI):** Con el objetivo de elevar los estándares de seguridad, escalabilidad y gestión del ciclo de vida del software, se proyecta la migración total de la arquitectura hacia **Google Cloud Platform (GCP)**. Esta evolución contempla:
* **Frontend y Seguridad:** Despliegue de la interfaz en **Firebase Hosting** o **Cloudflare Pages** para garantizar alta disponibilidad y protección contra ataques (DDoS/WAF).
* **Inteligencia Artificial (MLOps):** Migración del modelo *Random Forest* hacia **Vertex AI**. Esto permitirá automatizar los *pipelines* de reentrenamiento, gestionar el versionamiento de modelos y monitorear la degradación (*Model Drift*) en producción, profesionalizando la operación de la inteligencia artificial.

Bibliografía

Breiman, L. (2001). Random Forests. *Machine Learning, 45*, 5-32.

Django Software Foundation. (2023). Obtenido de Django Documentation: https://docs.djangoproject.com/

Fielding, R. T. (2000). *Architectural Styles and the Design of Network-based Software Architectures.* Irvine: University of California.

Gold Fields. (2024). *Descripción del entorno operativo Faena La Coipa.* Santiago: Gold Fields Salares Norte.

International Organization for Standardization. (2015). *ISO 9001:2015 Quality management systems — Requirements.* Geneva: ISO.

International Organization for Standardization. (2018). *ISO 45001:2018 Occupational health and safety management systems.* Geneva: ISO.

ISO. (2018). *ISO 45001:2018 Sistemas de gestión de la seguridad y salud en el trabajo.*

Jones, M., Bradley, J., & Sakimura, N. (2015). *JSON Web Token (JWT) - RFC 7519.* Fremont: Internet Engineering Task Force (IETF).

Merkel, D. (2014). Docker: lightweight linux containers for consistent development and deployment. *Linux Journal, 239*, 2.

Meta Platforms. (2023). Obtenido de React Documentation: https://react.dev/

Ministerio de Justicia. (1999). *Ley N° 19.628: Sobre Protección de la Vida Privada.* Santiago: Diario Oficial de la República de Chile.

Ministerio del Trabajo y Previsión Social. (2024). *Código del Trabajo: Artículo 184 (Protección de los trabajadores).* Santiago: Biblioteca del Congreso Nacional de Chile.

Mobley, K. (2002). *An Introduction to Predictive Maintenance.* Amsterdam: Butterworth-Heinemann.

Moubray, J. (1997). *RCM II: Reliability-centered Maintenance.*

Object Management Group. (2011). *Business Process Model and Notation (BPMN) Version 2.0.* Needham: OMG.

Open Source Initiative. (2024). Obtenido de The MIT License: https://opensource.org/licenses/MIT

PostgreSQL Global Development Group. (2024). Obtenido de PostgreSQL Documentation: https://www.postgresql.org/docs/

Redis Ltd. (2024). Obtenido de Redis Documentation: https://redis.io/docs/

Russell, S., & Norvig, P. (2016). *Artificial Intelligence: A Modern Approach* (3ra ed.). Essex: Pearson.

SAE International. (1999). *JA1011: Evaluation Criteria for Reliability-Centered Maintenance (RCM) Processes.*

Schein, A., & Popescul, A. (2002). Methods and metrics for cold-start recommendations. *Proceedings of the 25th Annual International ACM SIGIR*, (págs. 253-260).

Schein, A., Popescul, A., Ungar, L., & Pennock, D. (2002). Methods and metrics for cold-start recommendations. *Proceedings of the 25th Annual International ACM SIGIR*, (págs. 253-260).

Schwab, K. (2016). *The Fourth Industrial Revolution.* New York: Crown Business.

Schwaber, K., & Sutherland, J. (2020). *The Scrum Guide: The Definitive Guide to Scrum: The Rules of the Game.* Boston: Scrum.org.

SOMACOR S.A. (2024). *Informe de costos operacionales y multas Faena La Coipa.* Copiapó: Documento Interno.

ANEXOS

1. Sistema de información geográfica [↑](#footnote-ref-1)
2. Software de programación…. [↑](#footnote-ref-2)
3. OTS: ordnes de trabajos [↑](#footnote-ref-3)