Universidad de Buenos Aires
Facultad de Ciencias Exactas y Naturales
Departamento de Computación

Experience Report: ReportsHub

Tesis de Licenciatura en Ciencias de la Computación

Pablo S. Casullo

Director: Diego Garbervetsky

Buenos Aires, 2025

EXPERIENCE REPORT: REPORTSHUB

En las grandes corporaciones multinacionales coexisten múltiples tecnologías y prácticas de Business Intelligence (BI) que abarcan todo el espectro del stack tecnológico, desde el Data Warehouse (DW), pasando por los procesos de Extract, Transform and Load (ETL) [5] hasta herramientas de visualización y análisis. A pesar de los intentos por establecer estándares tecnológicos y de Gobierno [6], la descentralización de iniciativas entre áreas funcionales y niveles geográficos genera un ecosistema complejo, heterogéneo y difícil de armonizar. Este escenario plantea importantes desafíos tanto en la integración de soluciones como en la medición de su impacto en el negocio, donde la relación entre adopción e valor resulta poco evidente[2]. En este contexto, definir Key Performance Indicators (KPIs) de adopción se vuelve crítico para gestionar de manera eficiente el portafolio de soluciones de BI, aunque su implementación requiere superar barreras metodológicas y organizacionales. Asimismo, la User Experience (UX) se ve afectada por la falta de consistencia en el manejo de los accesos y servicios, lo que refuerza la necesidad de establecer enfoques más integrados y estandarizados en la gestión de estas herramientas.

Estas corporaciones, suelen tener prácticas de definiciones de estándares tecnológicos, que definen herramientas o tecnologías oficiales a ciertos productos y a proveedores, con el fin de simplificar y lograr sinergías tecnológicas entre las herramientas seleccionadas y reducir las combinaciones, según distintos casos de uso[7].

Palabras claves: *BI, DW, Stack tecnológico, UX, Portfolio management, Automatización, Decentralized Access Management, Global process, Gobierno.*

AGRADECIMIENTOS

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Fusce sapien ipsum, aliquet eget convallis at, adipiscing non odio. Donec porttitor tincidunt cursus. In tellus dui, varius sed scelerisque faucibus, sagittis non magna. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Mauris et luctus justo. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Mauris sit amet purus massa, sed sodales justo. Mauris id mi sed orci porttitor dictum. Donec vitae mi non leo consectetur tempus vel et sapien. Curabitur enim quam, sollicitudin id iaculis id, congue euismod diam. Sed in eros nec urna lacinia porttitor ut vitae nulla. Ut mattis, erat et laoreet feugiat, lacus urna hendrerit nisi, at tincidunt dui justo at felis. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Ut iaculis euismod magna et consequat. Mauris eu augue in ipsum elementum dictum. Sed accumsan, velit vel vehicula dignissim, nibh tellus consequat metus, vel fringilla neque dolor in dolor. Aliquam ac justo ut lectus iaculis pharetra vitae sed turpis. Aliquam pulvinar lorem vel ipsum auctor et hendrerit nisl molestie. Donec id felis nec ante placerat vehicula. Sed lacus risus, aliquet vel facilisis eu, placerat vitae augue.

A mi persona favorita.

Índice general

1. MOTIVACION

1.1. Contexto organizacional

En las grandes corporaciones multinacionales, existen múltiples tecnologías para resolver distintos desafíos de análisis e inteligencia de negocios, a lo largo de todo el stack tecnológico que va desde los sistemas ETL [5], distintos repositorios de almacenamiento para datos estructurados y no estructurados, y herramientas de exploración y explotación de datos que se asocian a la capa de presentación. En esta última coexisten planillas de cálculo (xls) y software específico de visualización para crear reportes y tableros de información, tales como Microstrategy, Cognos, PowerBI, Tableau o Qlik, entre otros.

Estas corporaciones suelen definir estándares tecnológicos que determinan herramientas o proveedores oficiales, con el fin de simplificar y lograr sinergias entre las tecnologías seleccionadas y reducir las combinaciones según distintos casos de uso [7].

Adicionalmente, es común que las iniciativas de inteligencia de negocios estén descentralizadas y distribuidas tanto en áreas funcionales (marketing, ventas, compras, finanzas), como en distintos niveles geográficos, que abarcan desde equipos corporativos hasta instancias globales, regionales y locales.

1.2. Situación inicial y limitaciones detectadas

Este contexto de distribución de recursos en áreas y geografías, combinado con la multiplicidad de herramientas "aprobadas" por dichas corporaciones, genera una combinación de componentes para la creación de distintas soluciones que, aun siguiendo las mejores prácticas de gobernanza tecnológica y de arquitectura [6], devienen en un ecosistema complejo, desarmonizado y lleno de particularidades.

Poder establecer el valor real e impacto que estas soluciones tienen en el negocio es un desafío, ya que en muchos casos la relación entre adopción e impacto es indirecta y no trivial [2]. A su vez, la experiencia de los distintos usuarios varía significativamente: solicitar acceso o encontrar los enlaces de cada herramienta resulta inconsistente, incoherente y dependiente de qué equipo haya desarrollado la solución.

1.3. Problemas principales para usuarios y equipos de datos

1.3.1. Gestión de accesos

La gestión de accesos junto con la aplicación de políticas de seguridad una vez otorgados los mismos, variaban absolutamente y quedaban definidas por el criterio de cada equipo que desarrollaba los contenidos, sin haber coherencia alguna. Algunos ejemplos de variantes incluyen:

Enviar un email a quien dio las capacitaciones (en caso que las haya habido). Muchas veces la persona encargada de la instrucción o entrenamiento de usuarios era también la encargada de controlar el acceso y otorgarlo. En otros casos, actuaba como intermediario para poder llegar a quien era responsable de otorgar los accesos adecuados. Solicitar acceso mediante herramientas de tickets a equipos encargados de la operación de dichos tableros/reportes. En equipos de proyecto de tamaños medio o grandes (con un número

de integrantes excediendo la decena), es común que haya gente dedicada exclusivamente a tareas operativas, cuyo objetivo es por un lado garantizar la continuidad y máxima disponibilidad de las soluciones como así también en ocasiones se encargan de ejecutar las actividades que habilitan acceso y perfiles de seguridad adecuados a los usuarios. Solicitar acceso a alguien conocido si puede hacer de nexo para dar con el contacto indicado.

1.3.2. Consumo de contenidos

Para poder consumir o utilizar reportes es fundamental saber de su existencia, conocer su ubicación (normalmente son enlaces dentro de redes internas) y tener acceso a los mismos. Algunos ejemplos acerca de cómo acceder, pueden incluir:

1. Usuarios reciben enlaces de acceso por emails o en documentos de entrenamientos y luego (con suerte) los almacenan como atajos en sus navegadores (con los problemas de tener hardcoded links que luego con el tiempo pueden variar, apuntar a versiones obsoletas de dichos reportes o incluso quedar deprecados conforme algunos reportes son decomisionados).

2. Páginas de intranet donde se publican los puntos de acceso (muchas veces implementadas como portales de acceso, provistos por las mismas herramientas de visualización).

3. Usuarios reciben reportes como archivos adjuntos en correos electrónicos.

1.3.3. Manejo del portafolio de soluciones de inteligencia de negocios

Para poder hacer un manejo efectivo de un portafolio de soluciones y de las tecnologías que se utilizan, es necesario poder entender:

¿Qué soluciones hay disponibles y en qué tecnologías están desarrolladas (inventario)?
¿Qué áreas de negocio cubren y quienes son responsables?
¿Qué nivel de adopción tienen?
¿Qué usuarios tienen acceso y no deberían?
¿Qué usuarios necesitan acceso y no tienen?
¿Qué usuarios tienen acceso y no lo usan?
¿Qué usuarios tienen altos nivel de adopción?
¿Qué volumen de pedidos de acceso manejan, según las audiencias objetivo?
¿Hay duplicidad de contenidos, hechos por áreas afines pero sin colaborar?
¿Hay patrones de uso de reportes que tengan correlación con el desempeño de áreas de negocio?

La respuesta a cada una y en particular a todas estas preguntas, en el entorno descripto, es de un esfuerzo que no permite tener información en tiempo y forma ni de modo adecuado, repetible y consistente de manera constante. Poder responder cada pregunta en un contexto tan heterogéneo, implica un nivel de trabajo manual, armonización y alineación de criterios y definiciones que simplemente lo vuelven imposible, en la escala de estas organizaciones.

En resumen, los impactos negativos descriptos anteriormente pueden sintetizarse en la siguiente lista:

- Falta de un repositorio central y común donde buscar/encontrar y solicitar acceso a los contenidos necesarios genera dificultad para poder dar con los contenidos.
- Procesos inconsistentes y altamente variables.
- Falta de transparencia en cuanto a puntos de contactos de reportes.
- Métodos de aprobación de múltiples pasos y manuales, que dependen de horarios laborales, feriados distribuidos en varias zonas horarios y múltiples geografías.
- Separación entre nivel de aprobación (hecha por la persona responsable) de un acceso y el nivel de ejecución de dicho acceso (hecha por operadores, una vez que la persona responsable ha definido qué acceso corresponde).
- Falta de información necesaria para el aprobador, acerca de rol, país, función y otros elementos que ayudan a determinar si un acceso debe o no ser otorgado a un usuario.
- La ausencia de un repositorio centralizado agrava los problemas de consistencia y coherencia. Esto se traduce en dificultades para asegurar estándares de seguridad homogéneos, en la duplicación de esfuerzos.
- El estado fragmentado limita la capacidad de contar con indicadores confiables y oportunos para la toma de decisiones de portafolio.
- La falta de métricas unificadas de adopción impide evaluar el impacto de las soluciones desarrolladas, dificultando la gestión adecuada del portafolio de inteligencia de negocios.

1.4. Aporte de la Tesis

Esta *tesis*, en formato de Experience Report, tiene como objetivo, profundizar en el proceso de desarrollo de una solución que atiende a los desafíos mencionados, detallando desde su concepción en el contexto organizacional, hasta su despliegue y medición del impacto, dentro de una organización de estas características.

Los contenidos a desarrollar incluyen:

Definiciones Preliminares: Conjunto de definiciones básicas que nos introducen en el dominio del problema.

Propuesta: Descripción detallada conceptual de la solución, en función de sus requerimientos funcionales y no funcionales. [12] [10]

Arquitectura: Descripción de los componentes y sus relaciones como también algunos patrones de diseño aplicados. [11] [3] [4]

Desarrollo y Despliegue: Explicación de la metodología aplicada y entregas iterativas.

Evaluación: Monitoreo y evaluación de las métricas clave y factores de éxito.

Lecciones aprendidas: Hallazgos y aprendizajes de la experiencia en el proyecto.

Conclusiones: Síntesis final del trabajo, con implicancias finales.

2. DEFINICIONES PRELIMINARES

Las definiciones preliminares son aquellas que nos permiten profundizar en los elementos básicos del dominio de nuestro contexto. A continuación se presentan las que aplican a conceptos, roles y elementos clave utilizados a lo largo de este trabajo.

2.1. Definiciones de industria y corporaciones multinacionales

KPIs: Key Performance Indicators (Indicadores clave de desempeño).

2.2. Definiciones de inteligencia de negocios

Inteligencia de negocios: (Business intelligence)

Data Warehouse:

2.3. Definiciones de diseño y arquitectura de software

Stack tecnológico:

Web App:

RLS:

LDAP:

API:

SOA:

Cohesión:

Acoplamiento:

Web Server:

Load Balancer:

Web Application Server:

Back end:

Client Server:

Caché:

Tolerancia a fallas:

NFS:

Database Server:

MVC: Model View Controller [8]. Patrón de diseño mediante el cual, se separan en tres capas bien definidas, para mayor flexibilidad, mantenibilidad y escalabilidad.

2.4. Definiciones de metodologías

Waterfall/Cascada: Descripción de la Metodología waterfall.

Agile/Agil: Descripción de la Metodología ágil.

DevOps: Metodología DevOps.

Unit tests: Pruebas unitarias.

Integration tests: Pruebas de integración.

Dev: Desarrollo. Entorno para desarrollo y pruebas de unidad y de integración.

UAT: User Acceptance Test. Entorno de pruebas de usuario y validaciones finales antes de hacer pasajes a producción.

Pr: Production/Producción. Entorno de operaciones productivo.

2.5. Definiciones propias del proyecto

ReportHub: Plataforma propuesta para consolidar y armonizar la publicación de reportes y tableros, gestión de accesos y monitoreo de métricas de uso de BI.

Administradores: Usuarios con privilegios para gestionar contenidos, accesos y usuarios dentro de la plataforma. Se distinguen tres tipos:

- **Admin Global:** Gestiona unidades geográficas globales y áreas de información a nivel global.
- **Admin Local:** Gestiona áreas de información dentro de su región geográfica asignada.

Creadores de Contenidos: Usuarios responsables de generar y publicar contenidos dentro de la plataforma.

Consumidores de Contenidos: Usuarios que acceden y utilizan los contenidos publicados en el portal, pudiendo también marcar favoritos y realizar solicitudes de acceso.

Aprobadores: Usuarios que evalúan y aprueban solicitudes de acceso a contenidos según la configuración de seguridad definida.

Estructura de contenidos: Jerarquía de organización de contenidos en dos niveles:

1. **Nivel 1:** Unidades geográficas (Global, Europa, América, Asia, África y países correspondientes).
2. **Nivel 2:** Áreas de información (temáticas específicas, únicas dentro de cada región).

Metadatos de Contenido: Información asociada a cada contenido publicado, incluyendo:

- Título
- Descripción

- Imagen miniatura
- Tipo de contenido (archivo o URL)
- Segmentos de usuarios y objetivos de frecuencia de uso
- Aprobadores
- Configuración de provisión de acceso

2.6. Definiciones de diseño y arquitectura de software

Stack tecnológico: Conjunto de tecnologías y herramientas utilizadas para construir y ejecutar una aplicación o sistema stack tecnológico.

Web App: Aplicación que se ejecuta en un navegador web y accede a servicios a través de Internet web application.

RLS: Row Level Security (RLS), directrices o políticas que definen cómo se maneja y protege la información sensible.

LDAP: Lightweight Directory Access Protocol (LDAP), un protocolo de red para acceder a servicios de directorio de información.

API: Application Programming Interface (API), un conjunto de rutinas y protocolos de programación que permiten la creación de software.

SOA: SOA, un enfoque de diseño en el que tanto las aplicaciones como los componentes de las aplicaciones son services intercambiables.

Cohesión: La medida en que las funciones relacionadas se agrupan en un módulo o clase [11].

Acoplamiento: La dependencia entre módulos o componentes de software [11].

Web server: Servidor que sirve páginas web y gestiona las solicitudes HTTP.

Load balancer: Dispositivo de red que distribuye la carga de trabajo entre múltiples servidores.

Web Application Server: Servidor que gestiona aplicaciones web y proporciona servicios adicionales como escalado y seguridad.

Back end: Parte del sistema que maneja la lógica de negocio, el almacenamiento de datos y la interacción con el front end back end.

Client server: Modelo de arquitectura de red en el que los clientes solicitan servicios al servidor.

Cache: Componente de hardware o software que almacena copias temporales de datos para acceder a ellos más rápidamente.

Tolerancia a fallas: La capacidad de un sistema para continuar funcionando a pesar de errores o fallos.

NFS: Network File System (NFS), sistema de archivos central y compartido que permite a múltiples computadoras acceder a archivos de forma simultánea.

Database Server (DB Server): Un servidor que gestiona bases de datos y permite el acceso y la manipulación de datos.

MVC: Model View Controller [8]. Patrón de diseño mediante el cual, se separan en tres capas bien definidas, para mayor flexibilidad, mantenibilidad y escalabilidad.

2.7. Definiciones de metodologías

Waterfall/Cascada: Una metodología de desarrollo de software en la que el proceso se divide en fases secuenciales y se asume que cada fase se completa antes de pasar a la siguiente [9].

Agile/Agil: Una metodología de desarrollo de software que promueve la iteración y la colaboración, y que se adapta a los cambios a medida que el proyecto avanza.

DevOps: El término "DevOps"fue acuñado por Patrick Debois en 2009 para nombrar su conferencia "DevOpsDays", la cual se inspiró en la charla "10 despliegues por día"de John Allspaw y Paul Hammond en la Conferencia Velocity de 2009. El término se creó al combinar Development (desarrollo) y Ops (operaciones) para resolver la brecha entre los dos equipos.[1].

Unit tests: Pruebas unitarias.

Integration tests: Pruebas de integración.

Dev: Desarrollo. Entorno para desarrollo y pruebas de unidad y de integración.

UAT: User Acceptance Test. Entorno de pruebas de usuario y validaciones finales antes de hacer pasajes a producción.

Pr: Production/Producción. Entorno de operaciones productivo.

CI/CD: Continuous Integration/ Continuous Development. Técnicas de automatización que permiten optimizar el ciclo de commit, regression testing y deploy.

2.8. Definiciones propias del proyecto

ReportHub: Plataforma propuesta para consolidar y armonizar la publicación de reportes y tableros, gestión de accesos y monitoreo de métricas de uso de BI.

Administradores: Usuarios con privilegios para gestionar contenidos, accesos y usuarios dentro de la plataforma. Se distinguen tres tipos:

- **Admin Global:** Gestiona unidades geográficas globales y áreas de información a nivel global.
- **Admin Local:** Gestiona áreas de información dentro de su región geográfica asignada.

Creadores de Contenidos: Usuarios responsables de generar y publicar contenidos dentro de la plataforma.

Consumidores de Contenidos: Usuarios que acceden y utilizan los contenidos publicados en el portal, pudiendo también marcar favoritos y realizar solicitudes de acceso.

Aprobadores: Usuarios que evalúan y aprueban solicitudes de acceso a contenidos según la configuración de seguridad definida.

Estructura de contenidos: Jerarquía de organización de contenidos en dos niveles:

1. **Nivel 1:** Unidades geográficas (Global, Europa, América, Asia, África y países correspondientes).
2. **Nivel 2:** Áreas de información (temáticas específicas, únicas dentro de cada región).

Metadatos de Contenido: Información asociada a cada contenido publicado, incluyendo:

- Título
- Descripción
- Imagen miniatura
- Tipo de contenido (archivo o URL)
- Segmentos de usuarios y objetivos de frecuencia de uso
- Aprobadores
- Configuración de provisión de acceso

3. PROPUESTA

3.1. Visión y objetivos del proyecto

Por todo lo anteriormente descrito, es que surge este proyecto, impulsando un espacio de consolidación y armonización de publicación, gestión de accesos y monitoreo de métricas de uso para la adecuada gestión del portafolio de BI. Asimismo, unificar el acceso y ofrecer una experiencia homogénea y consistente, independiente de la tecnología de implementación, se presenta como una oportunidad para mejorar tanto la eficiencia de los equipos de datos como la satisfacción de los usuarios finales.

De la complejidad de estas problemáticas surgió la necesidad de lograr una solución que permita:

Concentrar en un solo lugar la oferta de reportes/tableros ofrecida por los distintos equipos. Permitir a los equipos que publican estos elementos, realizar una gestión de sus usuarios, incluyendo la asignación de roles si aplican restricciones de visibilidad de datos. Establecer objetivos de adopción y medirlos de modo consistente, de manera totalmente independiente a la tecnología de implementación que se haya utilizado. Darle a los potenciales usuarios una experiencia homogénea y consistente, de modo agnóstico a las tecnologías utilizadas mediante técnicas de ingeniería de software. A los usuarios, almacenar atajos o accesos directos "resilientes" que resistan cambios de URLs de reportes y a la vez conserven atributos de seguridad para que compartiendo los enlaces la seguridad se mantenga.

3.2. Principios de diseño de ReportHub

El diseño de la solución se basó en un conjunto de principios arquitectónicos que aseguraron no solo la cobertura de las necesidades funcionales del momento, sino también la capacidad de evolucionar en el tiempo. Estos principios tuvieron como propósito garantizar que la plataforma fuera escalable, permitiendo crecer en volumen de usuarios, contenidos y procesos sin comprometer el rendimiento. A su vez, aseguraron un mantenimiento eficiente, reduciendo la complejidad técnica y posibilitando la incorporación de nuevas funcionalidades de forma ágil y con bajo costo operativo. Otro eje fundamental fue la facilidad de uso, que permitió que cada rol interactuara con la solución de manera simple, intuitiva y orientada a sus tareas principales, evitando barreras de adopción. Asimismo, se estableció un marco que permitió cumplir con las normas internas de auditoría, control de calidad y gobierno de la información, garantizando la trazabilidad y responsabilidad en cada acción ejecutada dentro del sistema. Finalmente, los principios incorporaron las mejores prácticas en seguridad y estándares internacionales, asegurando la protección de datos, la correcta gestión de accesos y la interoperabilidad con diferentes tecnologías, lo que fortaleció la resiliencia y confiabilidad de la solución en contextos de negocio dinámicos y regulados.

La solución debe cumplir con los siguientes principios de diseño:

3.2.1. Web application

Será una aplicación web, que contará con una elementos de presentación el el browser, una capa de lógica de negocio en un servidor y se apoyará en una base de datos relacional

para almacenar la información y meta información necesaria para la configuración, uso y auditoría de actividades.

3.2.2. lean UX minimalista y defensiva

Experiencia de usuario simple e intuitiva para cada uno de los roles en que los usuarios operen la solución, ya que un mismo usuario, puede tener contenidos para publicar, ser administrador del sistema y eventualmente consumidor de contenidos publicados por otros usuarios:

- *Administradores, roles:* "Admin Global / Admin Local".
- *Creadores de Contenidos, rol:* "Creador".
- *Consumidores de contenidos, rol:* "Consumidor".

La interfaz, debe evitar que el usuario cargue información inválida mediante reglas de negocio intuitivas y ya incorporadas a la navegación y cuando esto nos sea posible, validará la información que el usuario ingrese y proveerá feedback inmediato en el momento de las interacciones para poder corregir problemas.

3.2.3. Tener una SOA

Ser agnóstico de las tecnologías en las que se han producido los contenidos que se publicarán minimizando el acoplamiento técnico y a la vez funcionando con una alta cohesión. Este punto permite garantizar la "interoperabilidadçon diferentes tecnologías y a la vez el soporte de procesos comunes, sin estar condicionados por las tecnologías de implementación.

3.2.4. Descentrado y Escalable

Permitir la descentralización de la gestión de contenidos, accesos y configuraciones de modo de poder asignar en distintos niveles y equipos organizacionales las facultades para el auto servicio de sus contenidos. Esto evita los cuellos de botella de estructuras centralizadas y fomenta que se tomen las decisiones adecuadas en cada lugar adecuado de la organización, siguiendo procesos consistentes garantizados por el sistema.

3.2.5. Auditable

Cada acción debe ser auditable, para poder cumplir con normas internas de auditorías de calidad de procesos y responsabilidad, en particular a la hora de administrar usuarios y accesos.

3.3. Casos de uso principales

Los principales casos de uso, tienen como objetivo detallar en un nivel conceptual, las funcionalidades básicas y escenarios que iba a soportar el sistema.

3.3.1. Creación de estructuras geográficas para organizar los contenidos (Admin Global).

Los contenidos del sistema debían estar organizados en una jerarquía de dos niveles. El primer nivel, compuesto de unidades geográficas, tenía como objetivo agrupar por geografía y a su vez, poder otorgar niveles de administración descentralizados a distintos Admins de geografías para administrar áreas de información. Se utilizan como valores válidos, los correspondientes a las distintas unidades geográficas: -Global -Europa -America -Asia -Africa y luego países Todos estarán agrupados en un mismo nivel.

El segundo nivel de organización son áreas de información y permiten una agrupación lógica de los contenidos en función de las temáticas que cubren, como por ejemplo finanzas, marketing, ventas, etc. No existe un listado predefinido de qué áreas de información existen a nivel local y pueden ser creadas libremente por los administradores a nivel geográfico. La única restricción: los nombres de las áreas en una región geográfica deben ser únicos.

Por ejemplo:

- Global
  - Marketing
  - Cumplimiento
  - Ventas
  - Finanzas
  - Recursos Humanos, etc.
- America
  - Marketing
  - Legales, etc.
- Argentina
  - Fuerza de Ventas
  - Finanzas
  - Oportunidades, etc.

3.3.2. Gestión de Usuarios.

La “creación” y administración de usuarios, consiste en poder registrar usuarios que ya son parte del directorio corporativo como usuarios de este sistema y asignarles roles de Admins globales, locales, de áreas de información y/o eventualmente como consumidores. Del directorio corporativo se importan los siguientes datos: handle de usuario de la compañía, nombre completo, dirección de correo electrónico.

Es importante aclarar que un Admin tiene la capacidad de crear secciones, contenidos y manejar usuarios, pero no necesariamente tiene acceso a los mismos contenidos que publica, ya que son aspectos guiados por diferentes criterios de seguridad y las personas responsables de los contenidos son quienes deben aprobar los accesos y decidir quienes tienen acceso. La forma en la que los permisos serán asignados de acuerdo a los roles se detalla en la tabla 3.1.

Función/rol	Admin global	Admin local	Creador	Aprobador
Crear áreas globales	✓			
Crear áreas locales	✓	✓		
Crear contenidos	✓	✓	✓	
Administrar usuarios	✓	✓	✓	✓
Asignar Roles [1]	✓	✓	✓	✓
Aprobar accesos			✓	✓

Tab. 3.1: Funciones por rol (✓)

Rol / puede asignar	Admin global	Admin local	Creador	Aprobador
Admin global	✓	✓	✓	✓
Admin local		✓	✓	✓
Creador			✓	✓
Aprobador				✓

Tab. 3.2: Asignaciones de roles válidas (✓)

Todos los usuarios pueden asignar a otros usuarios las siguientes reglas de la tabla 3.2:

Estos mecanismos designados, también aplican a la hora de modificar los privilegios de un usuario. En caso de que un usuario vaya a ser desafectado del sistema, debe haber siempre un usuario alternativo como administrador global/local o creador de área de información o contenido. Si en algún momento algún usuario deja de pertenecer a la compañía, y algún área quedara sin administradores, entonces el sistema asignará automáticamente a cargo de los elementos del portal "huérfanos" a quienes sean los responsables inmediatos superiores dentro del mismo, siguiendo la lógica de Aprobador - Creador - Admin Local y Admin Global en última instancia. Para Admins globales debe haber siempre al menos 2 y es parte de la configuración inicial del sistema.

3.3.3. Gestión de contenidos (Creador/Admin).

La publicación de contenidos se hará de modo que cada pieza de contenido estará representada por los siguientes metadatos obligatorios:

- Título: Un texto denomina al contenido.
- Descripción: Texto que da una breve descripción de lo que el contenido ilustra o representa.
- Imagen miniatura: una imagen que representa el contenido, pudiendo ser un logo, un pequeño screenshot o cualquier elemento visual que permita reconocer al contenido publicado.
- Tipo de Contenido: Los contenidos pueden ser de dos tipos, o bien archivos cargables o bien URLs a elementos que residan dentro de la intranet.
- Segmentos de usuarios: Debe haber al menos 1 segmento de usuario (segmento por defecto llamado usuarios generales) y se le deben asignar objetivos de frecuencia de uso, en función de una cantidad de veces por unidades de tiempo. Cantidad:

Un número natural mayor a cero. Frecuencia: deberá ser algún valor de esta lista: "diario, semanal, mensual, trimestral, semestral, anual"

De este modo, la idea es poder establecer los objetivos de adopción según perfiles de usuario y cómo se espera que estos utilicen los contenidos para poder ser eficientes en sus procesos de negocio.

- Aprobadores: puede agregarse el listado de aprobadores y por defecto quien crea podrá aprobar acceso.
- Selección del modo de provisión de acceso:
  - En caso que el tipo de Contenido sea un archivo, se aplica la lógica por defecto de acceso directo.
  - En caso de que el tipo de Contenido sea un enlace a un reporte en otra tecnología, se opta entre dos modelos:
    1. Un modelo de control de acceso integrado a un sistema de tickets, donde se debe especificar dentro de ese sistema qué grupo es el encargado de recibir el ticket y se configuran parámetros con los que se creará el ticket, según sean requeridos:
       - Nombre del reporte al que se pide acceso.
       - Nombre de quien solicita el acceso.
       - Handle de usuario.
       - Dirección de email.
       - Puesto en la compañía.
       - País de localización de quien lo pide.
    2. Si el reporte tiene un modelo de acceso basado en listas de distribución del directorio corporativo, entonces deben especificarse dichas listas y quedarán asociadas al contenido (y se pueden asociar a segmentos de usuarios).
  - Si tiene seguridad basada en roles dentro del reporte y de ser así, se definen dos mecanismos posibles:
    1. Listas de distribución especificas del directorio corporativo. En este caso se define qué listas del directorio corporativo están asociadas a este contenido, para que el aprobador pueda utilizarlas en el momento de la aprobación.
    2. Esquema propio de cada contenido con perfiles/roles particulares. En este caso, deben especificarse los end points y credenciales para poder integrar la administración.

3.3.4. Navegación del catálogo y solicitud de acceso (Consumidor).

El portal tiene dos modos de operación. Un primer modo predeterminado, que permite utilizar los contenidos asignados, disponibles y ordenados según las unidades geográficas y áreas de información. También hay una sección de "Favoritos" que contiene aquellos contenidos elegidos por el usuario como favoritos, que normalmente simplifican el acceso a los que son de uso frecuente. Esta la posibilidad de buscar contenidos por textos relacionados a los metadatos definidos en el caso de uso 4. El segundo modo, es el modo "catálogo", donde el usuario consumidor tendrá la posibilidad de ver por unidades geográficas y áreas de

información disponibles pero a los que no tiene acceso. Cada elemento podrá ser agregado a un "carrito de compras", y una vez terminada la selección de los elementos para solicitar acceso, se podrá hacer un pedido formal de acceso. También está la posibilidad de agregar un texto como solicitud, explicando para cada elemento (o de forma colectiva) por qué la persona necesita acceso para los elementos. Cuando el usuario completa la acción de pedir acceso a los elementos del carrito de compras, se dispararán los procesos de aprobación, otorgamiento de accesos y notificación según se hayan definido en cada elemento.

3.3.5. Gestión de accesos y permisos (Creador/Admin).

En el circuito de aprobaciones, quienes son aprobadores recibirán una notificación por email y también tendrán un ícono en el portal que les indicará que tienen "tareas pendientes". Tanto el email como el ícono, lleva a los aprobadores a la interfaz que les permite evaluar los pedidos en función de las solicitudes que recibieron y poder aprobarlos o rechazarlos de manera general o particular. En caso de rechazo de la solicitud, deben colocar el motivo y en ambos (tanto positivo como negativo) casos el resultado del proceso será informado al usuario solicitante, tanto por email, como una notificación en el portal. Adicionalmente, si el contenido al que se solicita acceso, está cargado en el portal, el acceso se otorga de modo directo. En caso de que el contenido publicado tenga asociada seguridad manejada por listas de distribución, se hará la asociación en ese momento y si además tiene habilitada la configuración de administración de perfiles de aplicación, se asignarán en el momento. Si por último, el contenido publicado, está implementado y operado bajo un modelo de soporte basado en tickets, con la información configurada en el momento de la creación del contenido en el portal, se generará un ticket mediante el equipo correspondiente, adjuntando la información necesaria para el alta y la aprobación del aprobador para que el equipo de mantenimiento tenga el respaldo necesario para poder documentar y accionar el pedido, de acuerdo a las normas de cumplimiento de la empresa. En este caso, la notificación que recibirá el usuario que solicitó acceso tiene un texto que le informa que su pedido fue aprobado y se creó el ticket nro XXX en su nombre, para que pueda darle seguimiento con el equipo de operaciones.

3.3.6. Navegacion y acceso a los contenidos.

Las acciones de navegación dentro del portal tienen como objetivo poder recorrer los contenidos y accederlos. Para acceder un contenido, bastará con clickear con el mouse sobre la el ícono que lo representa en el catálogo y esto desplegará un nuevo Frame del navegador de internet que será una dirección enmascarada al elemento en cuestión. Esto permitirá que el enlace se pueda guardar como acceso directo y a la vez compartir por el usuario con otras personas, sin exponer el link original y preservando el control de acceso primario. El efecto deseado es agregar un nivel de indirección de modo tal que al guardar el link como acceso directo, quienes publican los contenidos puedan modificar el link interno de acceso sin afectar los enlaces del portal. Finalmente, todas las acciones de búsqueda, navegación, gestión de usuarios y de accesos, serán registradas en una bitácora de eventos de modo tal de generar un historial de las transacciones ocurridas, para poder luego poder optimizar el funcionamiento del sitio, aplicar auditorías de cumplimento necesarias y a la vez obtener métricas que permitan medir los criterios de éxito del Portal.

3.3.7. Logs de actividades y monitoreo de estadísticas.

hay que explicar acá qué es lo que se requiere para poder guardar la info y las estadísticas.

3.4. Alcance inicial y entregables previstos

El alcance inicial comenzó con la implementación de los casos de uso para los roles de Administrador Global, Creador y Consumidor de contenidos. En una segunda fase se decidió incorporar al perfil de administrador Local para poder agregar una capa intermedia de administración que permita descentralizar y federar el gobierno de los contenidos de modo más eficiente. Se priorizó también la publicación de contenido global en una primera instancia y luego en etapas sucesivas, luego de la implementación del rol local, se comenzó a expandir a otras unidades geográficas con el abordaje a los equipos locales de múltiples áreas.

3.5. Definición de KPIs y métricas de éxito

Para poder medir el avance y el éxito del proyecto se identificaron métricas de varios tipos:

1. *Experiencia de usuario*: La primer métrica que se definió está asociada al tiempo de respuesta del portal durante las interacciones y en principio, lo que se buscó es que cada interacción, dure entre 300 y 500 milisegundos (sin incorporar el tiempo de latencia de red). Es decir que cada vez que el usuario disparaba una acción sobre el portal, el tiempo de respuesta combinado, desde que enviaba el estímulo hasta que recibía la respuesta completa (o el indicio de respuesta) no debía ser mayor a 1000 milisegundos. Cuando hablamos del indicio de respuesta, nos referimos a que a veces, por volumen de información que debe viajar desde el servidor de base de datos y web hasta el browser del usuario, simplemente no es posible, pero sí se puede comenzar a renderizar parcialmente o dar alguna indicación visual de que su solicitud fue hecha y que está en proceso de ser resuelta.

2. *Adopción*: Para poder medir la adopción efectiva de la herramienta se estableció como métrica la cantidad de piezas de contenido creadas, cantidad de usuarios creadores, cantidad de usuarios asignados y finalmente actividad asociada según los tipos de transacciones definidos en los casos de uso.

3. *Ahorros de tiempo y recursos y minimización de errores*:

   Para poder hacer una medición de estos elementos se decidió hacer seguimiento de cantidad de tickets generados para gestión de accesos a grupos de soporte. Cantidad de accesos asignados mediante listas de distribución.

4. *Métricas de Auditoría y Cumplimiento*: Para poder hacer una medición del cumplimento se decidió identificar qué nivel de cobertura de transacciones eran documentadas apuntando al 100 en tiempo real como objetivo.

4. ARQUITECTURA

4.1. Principios de diseño

Para poder cumplir adecuadamente con los requerimientos funcionales (descriptos anteriormente en los casos de uso Sec. 3.3), como así también de los no funcionales, se decidió implementar diferentes principios de diseño.

A continuación, un detalle de qué principios de diseño se eligieron para dar adecuado soporte.

Para poder responder a diseño escalable y descentralizado, se seleccionó una arquitectura de *Web application*, ya que el encarar el proyecto con este enfoque, en lugar de una arquitectura de cliente servidor tradicional con un cliente compilado, ejecutable e instalado en las máquinas clientes y el código del servidor corriendo en un server, permitió aprovechar el mantenimiento y escalabilidad propias de las arquitecturas de las aplicaciones web.

En escencia, siguen siendo cliente servidor, con la diferencia de que el código que se ejecuta en el cliente se descarga en el código web al que se accede desde el browswer de internet, de modo que al mantenimiento y el control de versiones, actualizaciones y mejoras se da de modo automático[1] en cada máquina cliente.

Adicionalmente, con un diseño de despliegue físico adecuado, se puede también escalar horizontalmente sin mayores esfuerzos y aumentar la tolerancia a fallas.

En el caso puntual de esta aplicación, se diseñaron dos configuraciones: una para entornos de desarrollo y pruebas y luego una para un entorno productivo, donde se tolera una carga de usuarios real y debe maximizarse la tolerancia a fallas, donde se agrega un tercer servidor web/de aplicaciones.

[1] En ciertos casos, es posible que haya código descargado en cachés locales, pero pueden invalidarse y actualizarse fácilmente como parte de las actualizaciones.

Caso de Uso	**Principio de diseño**
3.3.1-Gestión de áreas geográficas	descentralizado y escalable 3.2.4
3.3.1-Gestión de áreas de información	descentralizado y escalable 3.2.4
3.3.2-Gestión de usuarios	descentralizado y escalable 3.2.4
3.3.3-Gestión de contenidos	descentralizado y escalable 3.2.4
3.3.4-Navegación del catálogo y solicitud de acceso	webapp 3.2.1 y lean UX3.2.2
3.3.5-Gestión de accesos y permisos	descentralizado y escalable 3.2.4
3.3.6-Navegacion y acceso a los contenidos	webapp 3.2.1 y lean UX 3.2.2
3.3.7-Logs de actividades y monitoreo de estadísticas	auditable 3.2.5

Tab. 4.1: Mapeo de Casos de uso a principios de diseño

Finalmente y de acuerdo a los estándares tecnológicos de este proyecto, se utilizó el siguiente stack tecnológico:

- Loadbalancer de F5
- Webserver: Apache
- Application server: Tomcat (Java)
- RDBMS: Oracle
- sistema operativo de servers: RH Linux

A continuación, describimos el diagrama de despliegue físico que permitió implementar la arquitectura de webapp, con máxima tolerancia a fallas en sus componentes de mayor demanda con escalabilidad horizontal.

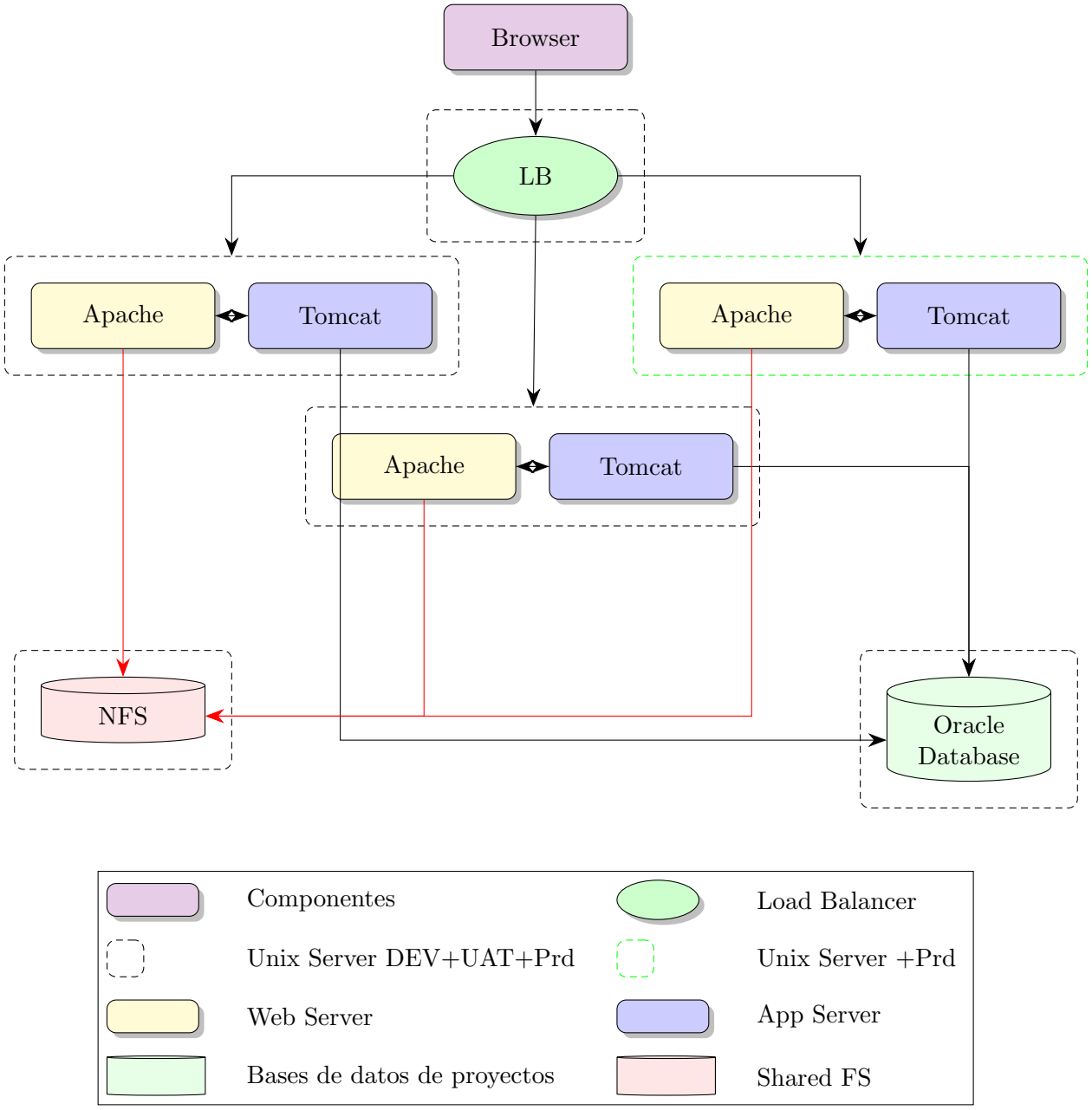


Fig. 4.1: Componentes de la arquitectura de la webapp - Diagrama de despliegue físico

1. Lean UX (3.2.2)

   En el caso del diseño de la interfaz de usuario, no condiciona la arquitectura en sí de la aplicación, aunque influye en la selección de qué tecnologías se utilizarán para implementar las interacciones y los elementos de visualización. La interfaz de usuario, es el sistema de comunicación que tiene el producto de software para brindarle información sobre el estado del sistema, los contenidos y el resultado de las operaciones que el usuario realice con el mismo.

2. Tener una SOA (3.2.3)

En este caso particular, se buscó implementar una arquitectura orientada a servicios, para la capa de negocio, de modo que pueda hacerse una separación, escalable de ser necesaria de ciertos servicios críticos y de alta demanda. de este modo, se puede seguir escalando la capacidad de la aplicación, en funcion de ciertas operaciones de alta demanda y críticas para brindar los servicios del portal. En otros casos, ciertos servicios se definieron como elementos reusables que pueden ser parte de un ecosistema más grande de aplciaciones y que tenía sentido independizar del bloque inicial de código, somo proyectos "hijos".

3. Descentralizado y escalable (3.2.4)

   La posibilidad de descentralización, tiene que ver con poder distribuir tareas según responsabilidades de ciertos usuarios, en vez de tener un equipo centralizado de gente que opere y administre los contenidos y los manejos de accesos del portal. Esto implica que el volumen de usuarios con la capacidad de ejercer múltiples roles aumenta, en prácticamente un orden de magnitud, de 10x a 100x.

   La escalabilidad, se vuelve entonces un aspecto escencial para poder sostener un buen desempeño, desde el punto de vista de la performance y la experiencia de usuario, sin poner en riesgo la estabilidad del sistema. Para dicho fin, la arquitectura de 4.1, mencionada anteriormente se vuelve clave.

4. Auditable (3.2.5) explicar toda la lógica de auditoría en función de qué transacciones que hayan identificar como auditables y luego empezar a desarrollar cada uno, los temas de metadata que deben registrarse. en un log de auditoría se registra:

   - userid: identificador unico del usuario que inicia la transacción.
   - timestamp (formato de timestamp que representa el instante de tiempo en el que se hizo la transacción con nivel de granularidad de milisegundo).
   - operación descripción de la operación que se realiza.
   - campos afectados (con valores anteriores y nuevos en caso de actualizaciones).
   - id de transacción (para el caso de que haya multiples operaciones en una transacción).
   - duracion (en milisegundos).

   Con toda esta metainformación, es posible obtener una vitácora de las transacciones ocurridas y poder realizar auditorías y análisis varios, tanto para evaluar estadísticas del sistema como así también patrones de uso y performance.

4.2. Arquitectura

4.3. Componentes principales

Para avanzar con la implementación de la solución, se decidió implementar un patrón de Model View Controller (MVC). Dicho enfoque, permite separar claramente en capas lógicas la presentación e interfaz, la lógica de negocio y el modelo de datos, que se alinean muy bien con la arquitectura de webapp.

El mapeo del modelo MVC a la arquitectura de webapp fue hecha del siguiente modo:
Modelo (Model):

En el contexto de una webapp, el modelo representa la parte de la aplicación que maneja los datos y la lógica de negocio. Incluye las clases y objetos que interactúan con la base de datos para almacenar, recuperar y manipular datos. El modelo se comunica directamente con el motor de bases de datos (RDBMS, Oracle, en este caso) para ejecutar consultas y obtener resultados. En esta arquitectura de múltiples servidores, el modelo está distribuido, con diferentes instancias de aplicación (Tomcat) accediendo a la misma base de datos compartida a través de un sistema de archivos compartido (NFS).

Vista (View):

La vista es la parte de la aplicación que el usuario interactúa directamente. En una webapp, es la interfaz de usuario presentada en el navegador del usuario. Las vistas se generan a partir del modelo y son dinámicas, adaptándose a los datos que el modelo proporciona. Las vistas pueden ser renderizadas en el servidor (Tomcat) y/o en el cliente (navegador). En este caso, las vistas son gestionadas por el servidor web (Apache), que sirve archivos estáticos (HTML, CSS, JavaScript albergados en el NFS) y delega las solicitudes dinámicas al servidor de aplicaciones (Tomcat).

Controlador (Controller):

El controlador es el intermediario entre la vista y el modelo. Es responsable de manejar las solicitudes del usuario, interpretar las acciones y actualizar la vista y/o el modelo en consecuencia. Aquí, el controlador se encarga de procesar las solicitudes HTTP, extraer los datos de la solicitud, invocar el código de negocio del modelo y devolver una vista actualizada al cliente. El controlador está implementado en el servidor de aplicaciones, donde maneja las interacciones con el usuario y coordina el flujo de datos entre la vista y el modelo.

A continuación, una breve explicación de cómo el MVC se mapeó a esta arquitectura de Web application:

Los componentes se distribuyen a lo largo de los diferentes niveles de la stack tecnológico:

- El servidor web (Apache) actúa como el punto de entrada para las solicitudes y atendiendo las solicitudes estáticas y delegando las dinámicas a los componentes tomcats.
- El servidor de aplicaciones (Tomcat) ejecuta el controlador y el modelo, procesando las solicitudes y generando las vistas.
- El sistema de archivos compartido proporciona el soporte al contenido estático que comparten las instancias de Apache y de Tomcats.
- La base de datos (Oracle) proporciona almacenamiento persistente para los datos generados, mantenidos y accedidos por el modelo.

A su vez, también, dentro del modelo, se identificaron los siguientes componentes lógicos que permitieron distribuir las actividades de desarrollo y mantener a la vez una estructura modular, de muy alta cohesión y bajo acomplamiento.

1. **ReportsHub**: Codigo que contiene la lógica de negocios y la capa de presentación que se sirve a los browsers con los que se accede.
2. **User Admin API**: Módulo de interacción con el directorio LDAP, que permite al portal ejecutar operaciones básicas de gestion de usuarios y de listas de distribución y grupos de seguridad.

3. **RLS API**: Módulo que implementa de manera genérica y universal, la gestión de perfiles de usuarios con el RLS que se utilice. Este módulo es de uso opcional y depende de un proyecto tiene implementado RLS o no.

   el conjunto de servicios implementados para dicho módulo es el siguiente:

   a) **GET_ROLES**: devuelve una lista de roles de proyecto.

   b) **GET_USER_ROLES**: devuelve el rol asociado a un usuario específico - usando un UserID.

   c) **ADD_USER_ROLES**: agrega o actualiza un rol a un usuario específico.

   d) **ADD_USER_W_FULL_NAME_ROLES**: agrega un rol a un usuario específico enviando el nombre completo también.

   e) **REMOVE_USER_ROLES**: elimina un rol de un usuario.

   f) **ADD_ACTIVITY_LOG**: envía información de auditoría al proyecto.

   Cada uno de estos servicios, se implementan como consultas a las bases de datos de cada proyecto, con las espeficidades de cada modelo de datos y sql, segun el rdbms que se utilice.

4. **RH DB**: Representa la instancia de la base de datos de ReportsHub, que almacena toda la información de catálogos, de usuarios del sistema, de actividad y de auditoría. En este caso puntual, una RDBMS Oracle.

Componentes del ecosistema:

Algunos elementos del ecosistema que debían utilizarse y con los que se tenía que interactuar con el directorio LDAP corporativo, para poder administrar usuarios y grupos de seguridad, bases de datos de los proyectos específicos y adicionalmente el sistema de autenticación de la corporación, para poder implmenentar Single Sign On (SSO)

1. LDAP Directorio corporativo.
2. Herramientas de visualización. [2]
3. Bases de datos de proyectos.

[2] En este caso se habla de herramientas de visualización porque son las relevantes para los proyectos de analytics, pero se puede generalizar a la gestión de accesos de cualquier tipo de software que pueda administrar sus accesos de usuarios apoyándose en listas de distribución o grupos de seguridad de LDAP

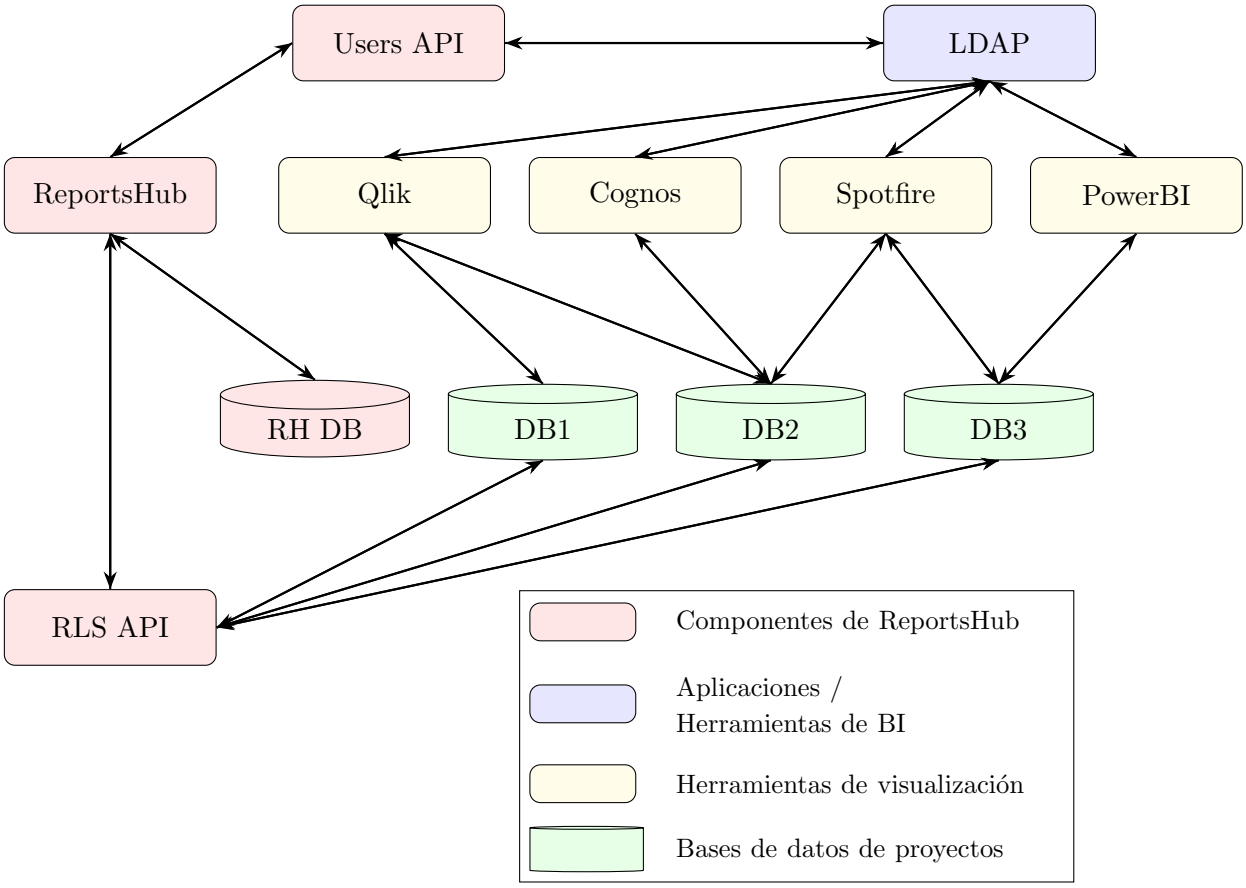


Fig. 4.2: Componentes de Arquitectura

4.4. Diseño de seguridad y Row Level Security (RLS)

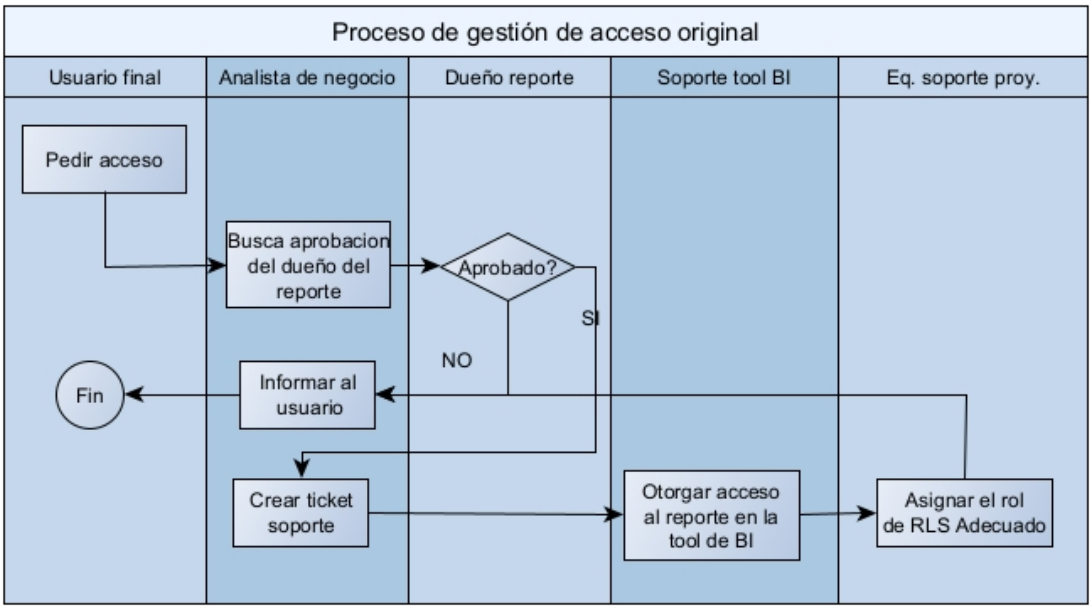
Por cuestiones de simplicidad y de diseño general, se trabajó con un esquema de seguridad donde usuarios de ciertas áreas funcionales, pueden tener acceso a la información completa de dichas áreas, con limitaciones geográficas. en el caso de los usuarios "locales", en general se les otorga permiso vingulado al país donde operan. usuarios regionales, se les otorga visibilidad sobre la región (información consolidada a nivel regional) y del detalle de los paises incluidos en dicha región. finalmente hay usuarios globales que tienen acceso a una capa global de datos (del area de interés del contenido publicado) y luego a los datos consolidados en niveles geográficos incluidos (regionales y paiese de cada región).

Normlamnete para implementar el filtrado a nivel geográfico, se aplicaron técnicas de RLS, de modo que permite mediante el rol del usuario, poder tener scope a las filas (rows) que son relevantes a dicho usuario.

La implmentación, normalmente se aplica a nivel de modelo de datos, haciéndola inherente a las consultas, de modo que siempre se agregan automáticamente a todas las queries, where conditions limitando las filas con las que se trabaja por alcance geográfico.

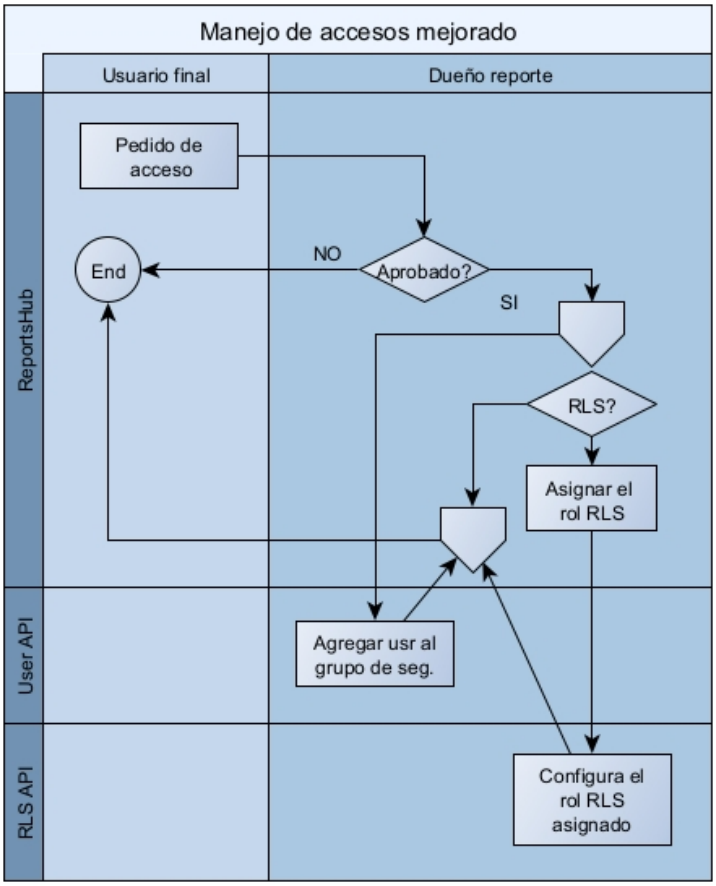
4.5. Mejora y simplificación en el proceso estándard de asignación de permisos

En el caso del proceso de asignación de permisos, puede observarse que intervienen dos roles, llamados Business Analysts y AMS BI Tool y AMS Project Support team que son resultado de la extrema burocracia y configuración sub-óptima del funcionamiento de las áreas de operaciones y mantenimiento.



El nuevo proceso resultante, quita del medio todos los elementos de çomplejidad accidental", dejando simplemente a los únicos dos roles, que son relevantes, el .[En]d User"que es el usuario final quien solicita el acceso al reporte y el Report Owner"que es quien determina si la solicitud corresponde o no y puede actuar en consecuencia.

EL proceso resultante de la simplificación e integración del portal conlas herramientas de visualización y también del RLS (en caso de ser necesario), se describe a continuación:


Manejo de accesos mejorado
Usuario final
Dueño reporte
ReportsHub
User API
RLS API
Pedido de acceso
End
NO
Aprobado?
SI
RLS?
Asignar el rol RLS
Agregar usr al grupo de seg.
Configura el rol RLS asignado

5. DESARROLLO Y DESPLIEGUE

5.1. Enfoque metodológico

El proyecto se estructuró de modo clásico Software Development Lifecycle (SDLC) con las siguientes fases:

1. Ideación: del concepto y problema general a resolver.
2. Especificación de requerimientos: Definicion de los requerimientos funcionales y no funcionales, junto con los criterios de aceptación.
3. Diseño de la solución: Definición de la arquitectura, de los componentes logicos, del modelo de datos y de la experiencia de usuario (mediante maquetas).
4. Desarrollo y pruebas: Fase de codificación y verificación de que el código cumple con los requerimientos y alcanza satisfactoriamente los criterios de aceptación.
5. Despliegue a producción y lanzamiento: Fase de despliegue en el entorno productivo y tareas de capacitación y entrenamiento de usuarios (y de equipo de soporte/operaciones)
6. Operación: fase en la que ya estando el sistema productivo, se le da soporte a los usuarios frente a incidentes.
7. Decomisado: fase en la que se planifica y ejecuta el decomisado de la aplicación una vez que ya ha cumplido su ciclo de vida (todavía no ha ocurrido esto, ni se espera que ocurra en los proximos 2-3 años)

Inicialmente se decidió trabajar con una metodología *Waterfall*, para poder enfocarse en un primer entregable, que pudiera cumplir con la funcionalidad básica de los casos de uso.

Para encarar el desarrollo, también se ensambló un equipo compuesto por los siguientes roles:

1. Product Owner: Persona encargada de definir el producto, su roadmap, prioridades y fases de entregas para maximizar el valor de cada entrega, en múltiples fases de desarrollo.
2. Project Manager: Persona encargada de armar el plan de trabajo con el product owner, los equipos de desarrollo, pruebas, operaciones y entrenamiento, para poder planificar las entregas y luego hacer la coordinar la ejecución y su seguimiento.
3. Equipo de desarrollo: Equipo responsable por escribir el código y realizar pruebas unitarias y verificaciones iniciales del código. Conformado por programadores especializados en visualización, en logica de negocio y bases de datos. También hubo apoyo part time de un arquitecto para poder trabajar las definiciones estructurales y detalles de arquitectura definidas en la sección anterior.

4. Equipo de testing: conformado por un líder de testing encargado de desarrollar los planes de pruebas, y desarrollar los casos de pruebas en funcion de las especificaciones de los casos de uso. Testers, encargados de ejecutar los casos de prueba, de distintos tipos, desde pruebas de integración, como así también de regresión y de performance.

5. Equipo de operaciones: Este equipo es el responsable de operar el sistema una vez desplegado en producción y a la vez de dar continuidad a la operación, en caso de que alguno de los componentes falle por algún problema. Frente alguna situación de falla o reportes de errores de usuario, son la primer capa de atención al usuario y que permite analizar el síntoma y hacer un diagnóstico, para saber si lo reportado por los usuarios es realmente un error del sistema, un problema de experiencia de usuario o de entrenamiento.

6. Equipo de capacitación: Equipo responsable de generar el material didáctico y planificar junto con el PM las capacitaciones de los usuarios, para poder garantizar una adopción adecuada del producto. También se explica el esquema de soporte en caso de necesitar reportar incidentes.

Rol / Fase	Ideación	Reqs	Diseño	Desarrollo	Despliegue	Operación
PO	**A / R**	**A / R**	C	C	C	I
PM	C	C	**R**	A	**A / R**	I
Architect	C	C	**A**	R	C	I
Desarrollo	I	C	R / C	**R**	C	I
Testing	I	C	C	**R / C**	C	I
Operaciones	I	I	C	I	**I**	**R / A**
Capacitación	I	A	I	I	**R**	C

Fig. 5.1: Matriz RACI-SDLC para el equipo definido

El esquema de trabajo en función de responsabilidades puede resumirse en la siguiente matriz Responsible Accountable Consult Inform (RACI)

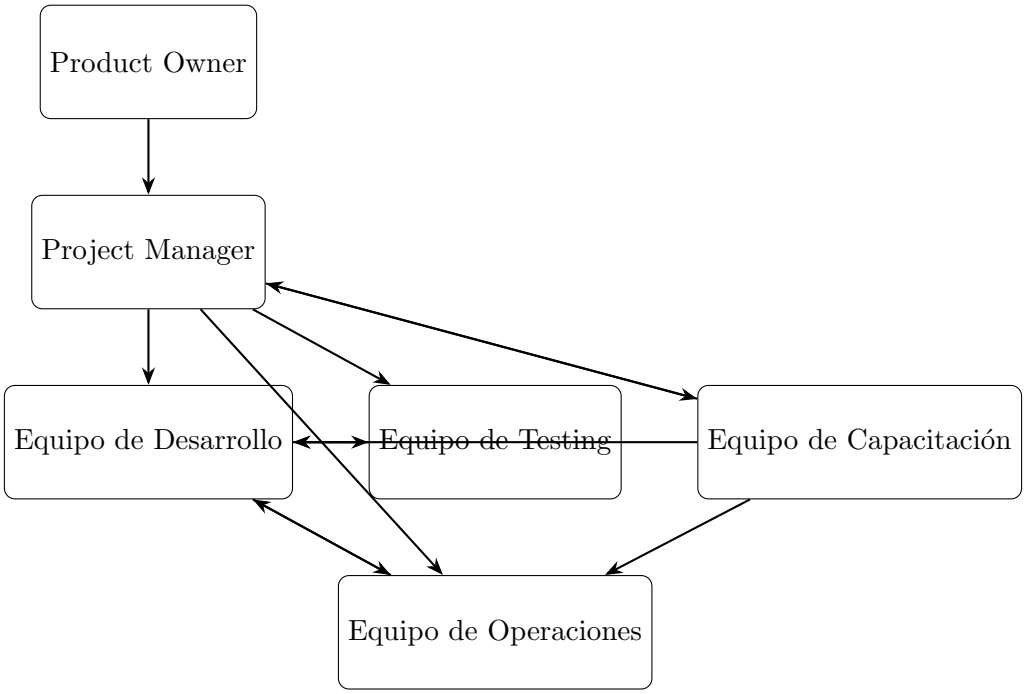
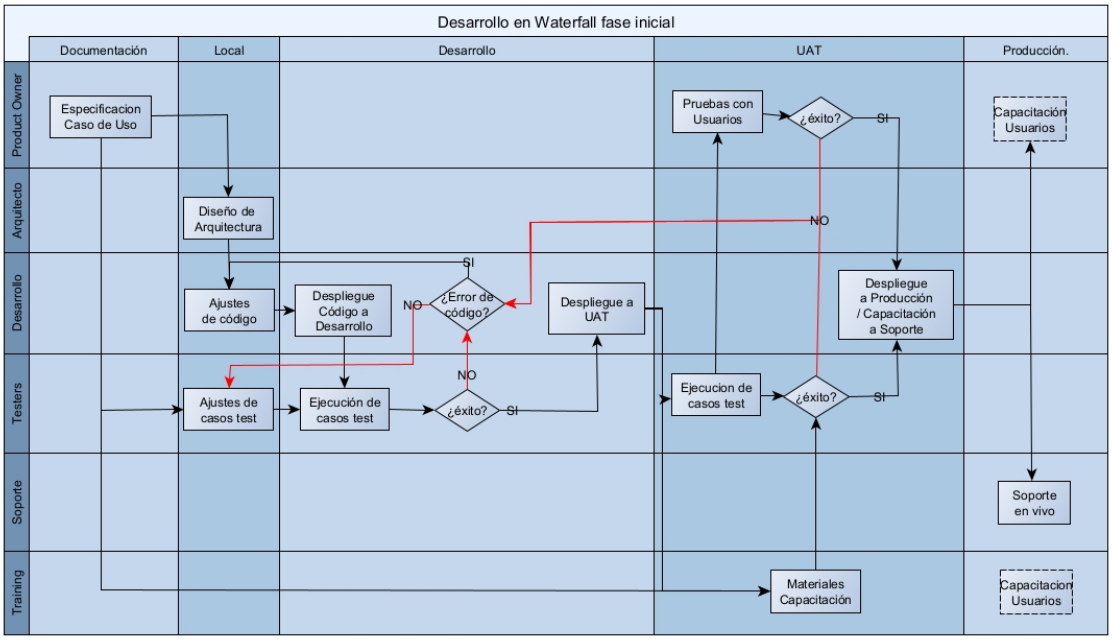


Fig. 5.2: Interacciones entre roles/sub-equipos

En esta metodología de desarrollo waterfall, se encuentran diferentes entornos de trabajo, dedicados a distintos fines y a la vez, a los que se accede, al cumplir satisfactoriamente diferentes verificaciones del ciclo de desarrollo.



1. Entornos locales: En los entornos locales, se trabaja de manera independiente en la documentación de casos de uso, diseño de arquitectura y luego cada desarrollador y tester, trabaja con versiones locales del codigo. En este caso, pueden exisistir

múltiples versiones del mismo código que se modifican en paralelo por los distintos desarrolladores, según se hayan dividido las tareas según los skills de los miembros del equipo. (puede ser front end, back end, modelos de datos, apis, etc.).

A su vez, el equipo de testing, en sus equipos locales, también desarrolla sus casos de prueba, alineados con la funcionalidad definida en la especificación, pero sin interactuar con el equipo de desarrollo, de modo que se toma como base la especifiación y no el código que desarrollan los programadores. De este modo, el equipo de testing se enfoca 100 % en la escencia de la funcionalidad y no necesariamente en cómo se ha desarrollado/implementado la funcionalidad.

2. Entorno de Dev: En el entorno de Dev. (o de Desarrollo), es el primero entorno en el que se realizan pruebas del código generado por los multiples desarrolladores, ya combinado en una versión candidata para ser lanzada a producción. Es el espacio donde el equipo de testing puede correr la totalidad de los casos de pruebas que generaron y se evalúan los resultados de las ejecuciones. Si son exitosas, en principio se habilita esa versión de la aplicación/código para ser promovido al entorno de UAT.

   En caso de resultar fallidas, el equipo de testing trabaja junto con el equipo de desarrollo para hacer el troubleshooting de aquellos casos fallidos y en función de los hallazgos:

   a) se corrije el código porque tenía defectos.

   b) se corrije caso de test porque tenía defectos.

   c) se corrijen el código y el caso de test porque ambos tenían defectos.

   Finalmente, se vuelve a unificar el código y los casos de pruebas ajustados para luego repetir las pruebas en el entorno de desarrollo hasta que todo sea satisfactorio.

3. Entorno de UAT: Este es el entorno donde el Product Owner, habilita acceso a ciertos usuarios clave, que ayudan a hacer una verificación del sistema para dar el visto bueno para poder pasar el sistema a producción. De surgir fallas, se reportan al equipo de testing y de desarrollo y se realiza el troubleshooting de modo similar al del estadio anterior. De este modo, cualquier falla que se detecta, lleva el proceso al ajuste de código y luego a las pruebas en el estadio anterior.

   Una vez que la aplicación se ha desplegado al entorno de UAT, las interfaces de usuario ya son práctiamente definitivas y el equipo de capacitación puede trabajar en el desarrollo de los materiales para la capacitación de los usuarios finales.

4. Entorno de Producción: Una vez que las pruebas en el entorno de UAT han sido superadas con éxito, se hace el pasaje final a producción y ocurren dos actividades escenciales. El product owner queda ya habilitado para comenzar las sesiones de capacitación según el plan de despliegue y entrenamiento y el equipo de desarrollo realiza una transferencia de conocimiento y capacitación (junto con la entrega de la documentación correspondiente) para que el equpo de operaciones, comience a dar el soporte necesario con la plataforma ya en vivo.

   El product owner, junto con el equipo de capacitación (training) hacen el lanzamiento y las campañas de comuniación/entrenamiento con la finalidad de entrenar a los usuarios en el uso del sistema y luego trabajar en la adopción del mismo.

5.2. Estrategia de despliegue y fases de rollout.

Para la estrategia de despliege, se optó, por un enfoque en dos fases. una Fase I, que consistía en un MVP con un pequeño grupo piloto de colaboración cercana, de modo de tener un primer acercamiento al uso real del sistema, con quienes se pudo trabajar muy de cerca. Esto permitió que se puedan testear las hipótesis y recolectar feedback valioso, para poder refinar los casos de uso originales.

Luego, se pasó a la fase II, fomentando un crecimiento .ºrgánico", donde se identificaron equipos clave en el resto de la organización y se fueron sumando gradualmente en un período de 12 meses.

5.2.1. Fase I: MVP con Grupo piloto

Para hacer el despliegue, se realizó la puesta en producción y se seleccionó un equipo de usuarios que trabajaba con la región de latinoamérica. Ellos tenían muy buena formación técnica y eran los clientes indicados para poder dar el puntapié inicial al despliegue global.

Se trabajó entonces con este equipo en modo piloto inicial y los resultados fueron alentadores.

Se realizaron las capacitaciones y luego estableció un período de tres meses para dejar rodar la solución y a la vez evaluar la estabilidad, performance y mejorar temas necesarios que no habían sido considerados como para poder luego hacer el rollout global.

5.2.2. Fase II: Expansión y escalamiento a más áreas y usuarios

Ya con el piloto satisfactorio, una lista mínima de mejoras ya implementadas y el equipo redefinido junto con la metodología, se decidió expandir el piloto al resto de las regiones y otros equipos corporativos. El despliege completo duró aproxiamdamente 6 meses más y a medida que se fue avanzando cada vez se fue simplificando más el proceso de .ºnboarding"de nuevos usuarios.

Se adoptó un enfoque de entrenamiento para los creadores de contenidos y ellos a su vez comenzaron a promocionarlo y entrenar a sus propios usuarios y en muchos casos, cuando había usuarios que ya usaban el portal por otros creadores de contenido, preguntaban proactivamente por qué faltaban agregar más contenidos, con lo cual el crecimiento se dio de modo orgánico y natural.

5.3. Plan de comunicación y capacitación

El plan de comunicación y de capacitación, se diseñó de modo de acompañar las fases de despliegue por grupos de usuarios y audiencias. Se encaró un enfoque de Community of Practice (CoP), que permitió crear un universo de usuarios general, con algunos más .ºexpertos"que ayudaron a disceminar las mejores prácticas.

se identificaron dichos usuarios líderes o clave y se hizo una capacitación del estilo Train The Trainers (TTT), para poder hacerlo en múltiples olas y a la vez de modo federado y escalable.

Los materiales de soporte y guías de usuario, se defineron en dos niveles:

1. Guía específica para usuarios con privilegios de publicación y administración de contenidos

2. Guía para usuarios finales, que en realidad estaba integrada al portal, e incluso la misma interfaz era bastante auto descriptiva.

5.4. Mecanismos de soporte post-lanzamiento

Dentro del esquema de soporte, se trabajó con un equipo corporativo que maneja un esquema de soporte de dos niveles. El primer nivel, orientado a preguntas simples y que son de solucion fácil, como pueden ser problemas de conectividad, instrucciones básicas sobre el uso del portal o eventualmente sobre alguno de los reportes. Si el primer nivel, no puede resolver satisfactoriamente la consulta del usuario, entonces se derivaba a un segundo nivel donde se identificaba, según la naturaleza de la consulta, qué acción seguir y qué grupo debía darle seguimiento.

Es importante destacar que el objetivo del equipo de nivel 2, es garantizar la continuidad del servicio y mantener el sistema disponible y los accesos de los usuarios activados.

En caso de que los usuarios finales hayan reportado un comportamiento compatible con un defecto o bug, entonces se derivaba al equipo de desarrollo para seguir con máxima prioridad un proceso de troubleshooting y análisis de causa raíz, para a su vez aplicar una solución definitiva.

5.5. Gestión de calidad y ajustes/mejoras metodológicas

Inicialmente, la calidad del desarrollo se fue monitoreando mediante el equipo de testing. EL leader de testing había creado un plan de pruebas y estas pruebas eran ejecutadas y documentadas rigurosamente antes de realizar pasajes a producción.

En caso de encontrar errores, se notificaban registraban en una plataforma que utilizabamos para tal fin y se compartían con el equipo de desarrollo y luego de corregirse, se re-verificaban.

El proceso era muy manual y no permitía, ni gran agilidad, ni múltiples y rápidas iteraciones a la hora de lanzar nuevas versiones a producción.

Después de un período inicial de desarrollo de 6 meses, se logró alcanzar el primer hito, donde se implementaron de manera básica todos los casos de uso definidos inicialmente.

Esto por un lado permitió ofrecer un MVP útil y funcional y a la vez, detectar oportunidades de mejora y evoluciones posibles, ya con el sistema rodando en el entorno de producción.

Para poder encarar sucesivas iteraciones en la evolución de la plataforma, se tuvo que cambiar drásticamente la composición del equipo, como así también la metodología de desarrollo, de modo tal de poder responder on mayor agilidad a los cambios y necesidades aumentando significativamente calidad y reduciendo los plazos de entrega.

Los cambios, fueron sustantivos y afectaron, no sólo la metodología de desarrollo, sino también la composición del equipo de desarrollo y operaciones.

Las principales causas para revisar la forma de trabajo fueron las siguientes: El esfuerzo manual de testeo, por parte del equipo de testing era excesivo, lento y un verdadero cuello de botella. La forma en la que se documentaban los defectos, a veces era inconsistente y se podían duplicar y a medida que la complejidad de la aplicación aumentaba, los casos de pruebas y sus tiempos de ejecución crecían de modo exponencial. La complejidad y diversidad casuística aumentaba a un ritmo mucho mayor que el código que la soportaba y era fundamental buscar alternativas.

Actividades	Waterfall	DevOps Interino	DevOps Final
Esquema de entregas	1 x Trimestre	1 x Mes	1-3 x Semana
Cobertura del código	≈ 25 %	≈ 25 %	≈ 50 %
Pruebas de Regresión (hs)	24	1	2
Tiempos de Hand Over (hs)	8	4	-
Tasa de defectos desde Prod (vs trabajo nuevo x mes)	≈ 30 %	≈ 15 %	≈ 5 %

Tab. 5.1: Mejoras metodológicas

Composición del Equipo	Waterfall	DevOps Interino	DevOps Final
ProductOwner	1	1	1
PM/ScrumMaster	1	1	1
Desarroladores	4	4	4
Arquitecto	0,3	0,3	0,3
DevOps Engineer	0	0,1	0,1
Lead de Test	1	1	0
Testers	4	3	0
Testers+	0	2	2
Training	0,3	0,3	0,3
Support Lead	1	0	0
Support	1	0	0
Total	13,9	13,4	8,4
Evolucion s %	100 %	≈ 96 %	≈ 60 %

Tab. 5.2: Evolución del Equipo

Esto además tenía un impacto muy negativo en los ciclos de entrega, haciendonos demorar meses hasta que una versión inicial y funcional, llegó a un punto de madurez que la hizo üsable".

Adicionalmente, había problemas en la calidad del código desarrollado por los desarrolladores, porque al demorarse las entregas, la presión por terminar aumentaba e impactaba los entregables generando retrabajo.

Finalmente, cuando llegaban las versiones al entorno de producción, notamos que en varias ocasiones, a pesar de haber pasado satisfactoriamente las fases de testeo y control de calidad, tenían defectos y el equipo de operaciones, tampoco era efectivo a la hora de resolver algunos incidentes, por falta de conocimiento de la aplicación, y las capacitaciones que el equipo de desarrollo les daba al hacer los despliegues productivos, eran complejas o insuficientes.

A continuación, se detalla cómo ha evolucionado la composición del equipo, a medida que se avanzó con la transformación metodológica, comparando Empleados de Tiempo Completo (ETC).

6. EVALUACION

Medir el éxito de este proyecto fue fundamental por varias razones:

1. Evaluación del rendimiento: Pudimos evaluar si los objetivos del proyecto se cumplieron y si los resultados esperados se alcanzaron. Esto ayudó a determinar si el proyecto fue efectivo y eficiente.

2. Identificación de áreas de mejora: Al medir el éxito, tambien pudimos identificar áreas que funcionaron bien y las que necesitan mejoras. Esto fue vital para aprender de la experiencia y aplicar esos aprendizajes a futuros proyectos.

3. Justificación de inversiones: Este proyecto requirió de una inversión, tanto en términos de tiempo como de dinero. La evaluación del éxito ayudó a justificar estas inversión, a demostrar el retorno sobre la inversión (ROI) a los interesados y sponsors y habilitar futuras fases de financiamiento para su evolución en el tiempo.

4. Motivación y reconocimiento: El haber medido y celebrado el impacto del proyecto, motivó al equipo del proyecto, reconociendo sus esfuerzos y logros. Esto mejoró la moral e imagen del equipo en la compañía y ayudó a fomentar una cultura de éxito y colaboración.

5. Mejora continua: La evaluación del éxito proporcionó datos y conocimiento que se utilizaron luego para mejorar los procesos y metodologías de gestión de proyectos. Esto contribuye a la mejora continua y a la optimización de futuras iniciativas.

6. Rendición de cuentas: Permitió al product owner rendir cuentas a los sponsors y stakeholders y demostrar que gestionaron los recursos de manera adecuada y responsable, cumpliendo con los objetivos establecidos.

A continuación, se detalla la metodología que se utilizó y los resultados obtenidos.

6.1. Metodología de evaluación

La metodología de evaluación se basó en tomar diferentes métricas y evaluarlas en función de las distintas fases de despliegue del proyecto.

Se evaluaron dichas métricas en un estado inicial (previo al despliegue del proyecto), como punto de partida, llamado *baseline* y luego, se tomó una primer evaluación, a los 12 meses de haber iniciado (y desplegado el proyecto) y se re-evaluaron dichas métricas para poder ver la evolución.

Las métricas que se evaluaron son las detalladas inicilamnete en la Sec.3.5 y rondan en tres ejes principales:

1. Métricas de experienca de usuario (1)

2. Métricas de adopción de la herramienta (2)

3. Métricas de ahorro y optimización (3)

4. Métricas de Auditoría y Cumplimiento (4)

6.2. Resultados cuantitativos

A continuación, se detallan las mediciones para cada metrica comparando el punto de partida (línea base) y el resultado al cabo de 12 meses del primer despliegue a producción.

ms por tx	Línea Base	12meses Prom	min	max
Gestion de Geografia 3.3.1	N/A	750	750	1020
Gestion de estructura 2 3.3.1	N/A	857	750	1020
Gestion de usuarios 3.3.2	N/A	1123	750	1020
Gestion de Acceso 3.3.5	N/A	950	750	1020
Navegacion 3.3.4	N/A	1050	750	1020
Gestion Contenido 3.3.6	N/A	600	750	1020

Tab. 6.1: Tiempos de respuesta en ms por transacción

Métricas de Adopción	Línea Base	12meses	Objetivo
Accesos Mensuales	0	7000(prom)/10000(max)	N/A
Reportes	0	307	550
Usuarios Únicos	0	1785	2500
Países	0	42	50

Tab. 6.2: Métricas de Adopción y Contenidos

Métricas de Ahorro y Optimizacion	Línea Base	12meses	Objetivo
Tickets de Soporte Creados Automáticamente	0	1036	N/A
Pedidos de Acceso	0	1447	N/A

Tab. 6.3: Métricas de Ahorro y Optimizacion

aquí, tengo que agregar un correlato entre el volumen de accesos y tickets y ahorros tanto de tiempos como eventualmente costos... (pendiente)

6.3. Resultados cualitativos

Adicionalmente, se hizo una medición çualitativa"del universo de usuarios, identificando aquellos de mayor peso en la organización y luego todo el resto.

Perfiles de Usuarios de Alto Impacto	Línea Base	12meses
Vice Presidentes Sr	0	2
Vice Presidentes	0	2
Vice Presidentes Asociados	0	2
Directores Ejecutivos	0	5
Directores	0	15
Otros	0	1759

Tab. 6.4: Perfiles de Usuarios de Alto Impacto

6.4. Impacto en seguridad y cumplimiento

Un objetivo clave del proyecto era abordar desafíos de cumplimiento que la organización enfrentaba debido a la falta de registros centralizados y mecanismos de auditoría eficientes. Para ello, el sistema de logs de auditoría automáticos y completos que permitió una captura detallada y precisa de todas las actividades relevantes.

El proyecto abordó estos desafíos mediante la implementación de un sistema de logs de auditoría automatizados, que incluía las siguientes características: 1. **Captura Automática de Datos**: Todas las actividades relevantes se registraban automáticamente, eliminando la necesidad de intervención manual y garantizando la integridad de los datos. 2. **Registros Completos y Detallados**: Los logs incluían información detallada sobre cada acción, incluyendo el usuario responsable, la fecha y hora, y la naturaleza de la acción. 3. **Acceso y Monitoreo en Tiempo Real**: Los responsables de cumplimiento podían acceder a los registros en tiempo real, permitiendo una supervisión continua y proactiva.

Beneficios Obtenidos: La implementación del sistema de logs de auditoría completos y automatizados resultó en varios beneficios para la organización:

- *Mejora en el Cumplimiento*: La capacidad de mantener registros detallados y accesibles permitió a la organización cumplir con las regulaciones de manera más efectiva y reducir el riesgo de sanciones.
- *Aumento de la Transparencia*: La disponibilidad de logs detallados mejoró la transparencia dentro de la organización, facilitando la rendición de cuentas y la toma de decisiones basada en datos.
- *Eficiencia Operativa*: La automatización de la captura de datos y la reducción de auditorías manuales permitieron una asignación más eficiente de recursos y una mejora en la eficiencia operativa.
- *Detección Temprana de Incidentes*: El acceso en tiempo real a los registros permitió la detección y respuesta rápida a incidentes, mejorando la seguridad y la gestión de riesgos.

El proyecto fue exitoso en resolver los desafíos de cumplimiento mediante la implementación de un sistema de logs de auditoría completos y automatizados. La organización no solo mejoró su capacidad de cumplir con las regulaciones, sino que también aumentó la transparencia, eficiencia operativa y capacidad de respuesta ante incidentes. Este enfoque proactivo y basado en datos ha fortalecido la posición de la organización en términos de cumplimiento y gestión de riesgos.

6.5. Conclusiones de la evaluación

El proyecto, luego de 12 meses, logró resultados destacados e impactos tangibles. siendo ahora nuevo estándard dentro de la corporación y permitiendo un nivel de simplicidad, acceso y eficiencia que parecia imposible. Una solución simple (aunque con cierta complejidad técnica) logró atender los desafíos propuestos en incluso ser adoptada globalmente.

7. CONCLUSIONES

7.1. Resumen de los logros principales

7.2. Relevancia del proyecto para la organización

7.3. Impacto estratégico a largo plazo

7.4. Próximos pasos sugeridos (roadmap evolutivo)

8. LECCIONES APRENDIDAS

8.1. Aspectos que funcionaron bien

8.2. Dificultades y cómo se superaron

8.3. Factores clave de éxito

8.4. Aspectos a mejorar

8.5. Aprendizajes organizacionales

8.6. Recomendaciones para futuros proyectos

BIBLIOGRAFíA

[1] John Allspaw y Paul Hammond. «10 deployments per day: Dev and ops cooperation at Flickr». En: *Proceedings of the 2009 conference on Velocity*. Santa Clara, CA: USENIX Association, 2009.

[2] Thomas H. Davenport. «Competing on analytics». En: *Harvard Business Review* 84.1 (2006), págs. 98-107.

[3] Thomas Erl. *Service-Oriented Architecture: Concepts, Technology, and Design*. Prentice Hall, 2005.

[4] Martin Fowler. *Patterns of Enterprise Application Architecture*. Addison Wesley, 2003.

[5] W. H. Inmon. *Building the Data Warehouse*. New York, NY: John Wiley & Sons, 1992.

[6] ISACA. *COBIT 5: A Business Framework for the Governance and Management of Enterprise IT*. Rolling Meadows, IL: ISACA, 2012.

[7] Ivar Jacobson. «Object-Oriented Development in an Industrial Environment». En: *Proceedings of the OOPSLA '87 Conference*. 1987.

[8] Glenn E. Krasner y Stephen T. Pope. «A Description of the Model-View-Controller User Interface Paradigm in the Smalltalk-80 System». En: *ParcPlace Systems, Inc.* (1988).

[9] Winston Royce. «Managing the Development of Large Software Systems». En: *Proceedings of IEEE WESCON*. Vol. 26. IEEE, 1970, págs. 1-9.

[10] Raymond T. Yeh y Pamela Zave. «Specifying software requirements». En: *Proceedings of the IEEE* 68.9 (1980), págs. 1077-1085.

[11] Edward Yourdon y Larry L. Constantine. *Structured Design: Fundamentals of a Discipline of Computer Program and Systems Design*. Prentice Hall, 1979.

[12] Pamela Zave. «A comprehensive approach to requirements problems». En: *Proceedings of the IEEE Computer Society's Third International Computer Software and Applications Conference (COMPSAC '79)*. IEEE, 1979, págs. 117-122.