Universidad Internacional de La Rioja
Escuela Superior de Ingeniería y Tecnología

Máster Universitario en Desarrollo y Operaciones (DevOps)

DataOpsHub: Plataforma Unificada para Gobernanza, Gestión y Orquestación de Artefactos Analíticos con SecDevOps

Trabajo fin de estudio presentado por:	Hector Jeancarlos Santos Nuesi, Alex Cayoja Perez
Tipo de trabajo:	Trabajo fin de Master
Director/a:	JOSEFINA GUERRERO GARCIA
Fecha:	22/10/2025

Resumen

En la actualidad, muchas organizaciones gestionan sus artefactos analíticos (modelos de machine learning, scripts y transformaciones de datos) de forma fragmentada. Esta carencia de mecanismos unificados de seguridad, control, versionamiento, integración y monitoreo genera duplicidad, riesgos operativos, alto costo computacional y dificultades de colaboración entre los equipos de datos, Business Intelligence y áreas funcionales.

El presente trabajo propone el diseño e implementación de DataOpsHub, una plataforma web integral orientada a automatizar, auditar y gobernar el ciclo de vida de artefactos analíticos mediante principios de SecDevOps y DataOps. Esta solución centraliza funciones críticas, como la ejecución, versionamiento, reentrenamiento y despliegue de modelos, además de gestionar permisos por rol y ejecutar pruebas unitarias, de integración y End-to-End (E2E). Incluye monitoreo continuo a través del stack Prometheus, Grafana y Loki.

Para su desarrollo, se aplicó una metodología ágil y SecDevOps, utilizando un stack tecnológico robusto: Angular para el frontend, Python para el backend, Jenkins para CI/CD, Docker para contenerización y Terraform para un despliegue cloud-agnostic. Se integraron análisis estáticos y pruebas automáticas para asegurar la calidad y segurida.

El resultado es una plataforma escalable, portable y gobernada que estandariza procesos analíticos, mejora la colaboración y aumenta la confiabilidad operativa. DataOpsHub demuestra un marco efectivo para unificar DataOps, MLOps y SecDevOps en entornos corporativos modernos.

Palabras clave: SecDevOps, DataOps, Gobernanza de datos, Automatización de artefactos analíticos, Monitoreo continuo.

Abstract

Modern data-driven organizations generate diverse analytical artifacts (including ML models, automated jobs, and data transformations) often managed in a fragmented and ungoverned manner. This lack of centralization creates operational risks, limits traceability, and reduces collaboration efficiency among data teams. To address these challenges, this project introduces DataOpsHub, a unified SecDevOps-based platform designed to automate, orchestrate, audit, and govern the full lifecycle of analytical artifacts. DataOpsHub centralizes control through secure, role-based authentication, automates ML model execution and retraining, manages job orchestration, and integrates unit and end-to-end testing. Additionally, it ensures real-time observability through an integrated Prometheus, Grafana, and Loki stack.

Keywords: SecDevOps, DataOps, Data Governance, Automation of Analytical Artifacts, Continuous Monitoring.

Índice de contenidos

Índice de figuras

Índice de tablas

1. Introducción

La adopción en expansión de enfoques basados en datos ha disparado la producción de activos clave (incluyendo modelos de ML, procesos ETL, dashboards y scripts) creando una demanda crítica por soluciones unificadas que permitan el control integral (gestión, orquestación, auditoría y compartición) de estos activos bajo criterios de seguridad, estandarización y escalabilidad.

En muchas organizaciones, especialmente aquellas con equipos de Ciencia de Datos, Ingeniería de Datos, Business Intelligence y áreas funcionales con data owners, gestionan sus artefactos y procesos de manera manual, carecen de repositorios estandarizados, ausencia de metadata clara, falta de versionamiento efectivo y poca trazabilidad operativa. Esto ocasiona fallos en la colaboración, pérdida de control, datos duplicados, modelos obsoletos y altos costos operativos debido a la falta de gobernanza.

El presente trabajo propone el diseño e implementación de DataOpsHub, una plataforma web modular orientada a la automatización, seguridad y gobernanza integral del ciclo de vida de los artefactos analíticos. Esta plataforma centralizará la ejecución, auditoría, reentrenamiento y despliegue de modelos de machine learning; la gestión de procesos automatizados; la ejecución de pruebas unitarias y de integración; el monitoreo en tiempo real; la autenticación segura con permisos por rol; y la capacidad de desplegar la solución por contenedores Docker en cualquier proveedor cloud usando Terraform.

El proyecto integra principios de SecDevOps, DataOps y MLOps, permitiendo que los artefactos creados por distintos equipos puedan ser gobernados, probados, desplegados y monitoreados bajo un framework unificado, basado en herramientas modernas como Angular, Python, Jenkins, Docker, Prometheus, Grafana, Loki y SonarQube.

1.1.Justificación del trabajo

Para operar eficazmente, las entidades corporativas contemporáneas requieren una infraestructura sólida que permita la gestión unificada de todos los artefactos generados por sus equipos de análisis, ingeniería y ciencia de datos. No obstante, la realidad operativa frecuentemente revela:

- Modelos entrenados sin versionamiento adecuado.
- Scripts y jobs críticos distribuidos de forma desordenada entre diferentes usuarios o departamentos.
- Ausencia de trazabilidad operacional que impide auditar ejecuciones, fallos y reentrenamientos.
- Falta de un mecanismo universal para compartir artefactos entre áreas.
- Inexistencia de una capa de seguridad que controle el acceso según perfiles, roles o dominios de datos.
- Escaso o nulo monitoreo de los procesos que dificultan detectar desviaciones, degradación de modelos o sobrecargas.

Este trabajo es relevante porque propone una plataforma capaz de resolver todas estas problemáticas mediante un enfoque holístico, integrando SecDevOps, DataOps y MLOps bajo un mismo ecosistema.

Además, tiene gran valor profesional porque:

- Permite estudiar la convergencia entre analítica, ciencias de datos y automatización.
- Aporta un marco aplicable a empresas reales que carecen de gobernanza de artefactos.
- Ofrece un enfoque escalable y portable a múltiples entornos cloud.
- Se fundamenta en tecnologías modernas, open-source y alineadas a buenas prácticas industriales.

1.2. Planteamiento del problema

En el entorno actual, los equipos de análisis trabajan de manera fragmentada, utilizando herramientas y metodologías distintas sin un punto central de gobierno y ejecución. Esto genera múltiples problemas, tales como:

- Inconsistencias entre versiones de modelos y scripts.
- Dificultad para compartir artefactos entre áreas o usuarios.
- Falta de mecanismos de aprobación, auditoría o pruebas previas al despliegue.
- Riesgos de seguridad al no existir control de accesos ni permisos individuales.
- Imposibilidad de monitorear el comportamiento operativo de artefactos críticos.
- Dependencia de procesos manuales o pipelines aislados.

Para abordar estas deudas técnicas, proponemos construir una plataforma unificada que permita:

- Centralizar todos los artefactos analíticos.
- Gobernar su acceso por roles, tokens y autenticación segura.
- Ejecutar, versionar y auditar sus acciones.
- Proveer re-entrenamientos de modelos bajo demanda o programados.
- Crear y orquestar jobs automatizados.
- Integrar pruebas unitarias, de integración y de calidad.
- Monitorear y alertar sobre estados críticos.
- Habilitar el despliegue bajo demanda de artefactos y procesos previamente probados y certificados para producción, utilizando infraestructura como código para asegurar consistencia y trazabilidad.

1.3. Estructura de la memoria

La presente memoria se organiza en cinco capítulos que describen de manera progresiva el contexto, diseño, desarrollo e impacto de la plataforma DataOpsHub.

Capítulo 1 introduce los elementos fundamentales del trabajo, exponiendo la justificación, el planteamiento del problema y una visión general del objetivo de la investigación. Además, presenta una descripción de la estructura del documento para guiar la lectura.

Capítulo 2 profundiza en el contexto teórico y en el estado del arte. Incluye una revisión de los antecedentes relacionados con SecDevOps, DataOps, MLOps, automatización, orquestación de procesos y gobernanza de artefactos analíticos. También analiza trabajos y soluciones existentes, identificando sus limitaciones y estableciendo el marco conceptual que sustenta la propuesta de DataOpsHub.

Capítulo 3 define el objetivo general, los objetivos específicos y la metodología utilizada. Se detalla el enfoque SecDevOps adoptado, las herramientas seleccionadas y las fases del proceso de desarrollo, sustentado en prácticas ágiles y en la integración continua de seguridad, pruebas y despliegue.

El Capítulo 4 describe el desarrollo específico de la contribución. Incluye el análisis de requisitos, la planificación del proyecto, el diseño de la arquitectura, la implementación del frontend y backend, los procesos de automatización, la integración del monitoreo y las pruebas. Asimismo, se presentan los mecanismos de evaluación, abarcando la evaluación de modelos, la evaluación de seguridad y otras pruebas fundamentales para validar la plataforma.

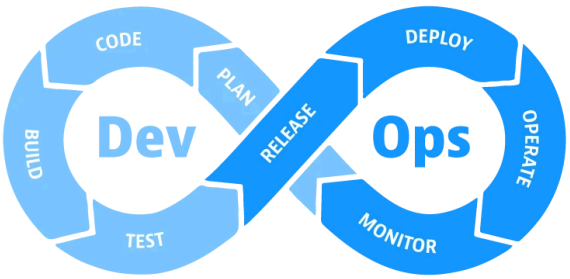
Capítulo 5 expone las conclusiones generales del trabajo, analizando el grado de cumplimiento de los objetivos planteados y el impacto de la solución propuesta. También se presentan posibles líneas de trabajo futuro orientadas a expandir las capacidades de la plataforma y fortalecer su aplicación en entornos empresariales reales.

2. Contexto y estado del arte

2.1. Contextualización y antecedentes

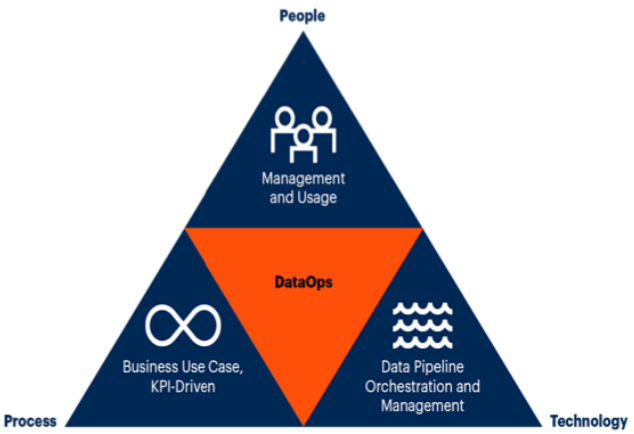
El paradigma DevOps surgió como una respuesta al tradicional aislamiento entre los equipos de desarrollo y operaciones, promoviendo integración continua, entrega continua y feedback permanente para mejorar la calidad del software y acortar los ciclos de entrega. En este sentido, "DevOps is more than a pure methodology … it represents a paradigm addressing social and technical issues in organizations engaged in software development". (Dominik Kreuzberger, Niklas Kühl, Sebastian Hirschl, 2022)

Figura 1: Ciclo DevOps.



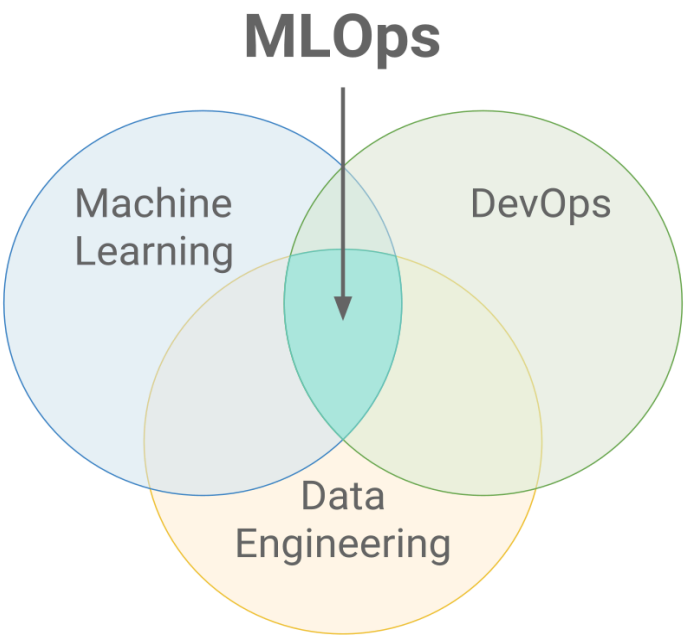
Con el crecimiento de los datos y la analítica, emergieron disciplinas como DataOps, que adaptan los principios de DevOps al dominio de los datos, haciendo énfasis en la automatización de pipelines de datos, la gobernanza, la calidad de los datos y la entrega ágil de valor desde las fuentes de datos hasta el análisis. (Sandhya Nayak, Sonia Mezzetta, Karina Kervin, 2021)

Figura 2: Dimensiones DataOps.



Por otra parte, el surgimiento de "Machine Learning Operations (MLOps)" responde a necesidades específicas del ciclo de vida de modelos de machine learning en producción: entrenamiento, deployment, monitorización, retraining y gobernanza de modelos. (PlaysDev, 2024)

Figura 3: Diagrama LMOps.



Estas disciplinas, junto con la infraestructura como código (IaC) y una creciente atención a la seguridad (DevSecOps), forman el ecosistema tecnológico y metodológico donde se inserta este trabajo, que busca aportar una plataforma integral para artefactos analíticos.

Figura 4: IaC separa los recursos de configuración del hardware o software para que puedan almacenarse, compartirse y administrarse como código.

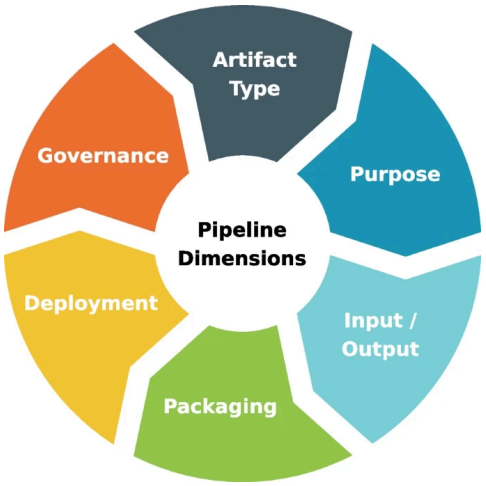
2.2. Trabajos relacionados

Una revisión sistemática reciente analiza cómo la integración de DataOps y MLOps permite despliegues escalables de machine learning, señalando que uno de los retos clave es la

gestión de datos, la automatización de pipelines y la complejidad operativa. (Bhanu Prakash Reddy Rella, 2022)

El estudio "ArtifactOps and ArtifactDL" propone una metodología que unifica DevOps, DataOps y MLOps bajo un mismo marco para tratar artefactos de código, modelos, datos e infraestructura, lo que evidencia la tendencia hacia soluciones más holísticas que la que tradicionalmente se usa en la industria.

Figura 5: Dimensiones de Pipelines definidas.



Asimismo, investigaciones enfocadas en DataOps señalan que la participación en ecosistemas de datos requiere prácticas ágiles y lean de gestión de datos, automatización de pipelines y calidad de datos como principio estratégico. (Inan Gür, 2021)

Sin embargo, la mayoría de plataformas existentes se centran en uno o dos dominios (por ejemplo modelo + datos, o datos + orquestación) y no cubren de manera integral la combinación de artefactos, pruebas automatizadas, gobernanza, orquestación de jobs, despliegue multi-cloud y control de acceso seguro. Esta laguna justifica la propuesta de este trabajo.

2.3. Conclusiones del estado del arte

Existe un consenso amplio en la literatura de que DevOps, DataOps y MLOps comparten principios comunes (automatización, CI/CD, monitorización, feedback continuo), pero que sus ámbitos de aplicación, herramientas y retos difieren.

Tabla 1: Comparativa de Metodologías de Operaciones (Ops).

Aspecto	**DevOps**	**DataOps**	**MLOps**	**AIOps**
Enfoque Principal	Desarrollo de software y automatización de operaciones.	Eficiencia del *pipeline* de datos y calidad de los datos.	Gestión del ciclo de vida de los modelos de *machine learning*.	Automatización inteligente de las operaciones de TI.
Tecnologías Clave	Jenkins, Git, Docker, Kubernetes, Ansible, JIRA, Chef	Apache Airflow, Databricks, DataKitchen, Highbyte	Python, TensorFlow, PyTorch, Jupyter Notebooks	Motores de IA/ML, plataformas de *Big Data*, herramientas de monitoreo de TI
Principios Fundamentales	CI/CD (Integración/Entrega Continua), automatización de infraestructura, colaboración	Orquestación de *pipelines* de datos, control de versiones para artefactos de datos	Entrenamiento, despliegue, monitoreo y versionamiento de modelos	Detección proactiva de incidentes, análisis de causa raíz y remediación

				automatizada
Usuarios Clave	Ingenieros de software, Ingenieros DevOps, equipos DevOps	Ingenieros de datos, profesionales DataOps	Científicos de datos, Ingenieros ML	Equipos de operaciones de TI, científicos de datos, ingenieros AIOps
Casos de Uso	Integración continua, contenerización y microservicios	Ingesta, transformación de datos y analítica en tiempo real	Analítica predictiva, *pipelines* ML y despliegue de IA	Respuesta automatizada a incidentes, optimización de infraestructura

Las soluciones que integran gobernanza, control de accesos, artefactos analíticos (modelos, scripts, procesos) y despliegue portátil en la nube aún son escasas o están fragmentadas.

Este panorama revela un vacío importante: las organizaciones necesitan una forma de gestionar de manera unificada todos sus artefactos analíticos, imponiendo buenas prácticas DevOps (automatización, pruebas, CI/CD) al tiempo que aseguran la gobernanza (seguridad, cumplimiento, linaje) y la operación estable en producción (monitorización, alertas).

La propuesta DataOpsHub nace precisamente para llenar ese vacío, integrando en una sola plataforma las capacidades dispersas en las herramientas actuales. En base a los hallazgos, DataOpsHub se justifica como una solución que combina: un portal central con autenticación

y roles (al estilo de un catálogo de datos seguro), un motor de orquestación para ejecutar pipelines de datos y jobs de ML (comparable a Airflow/Dagster pero consciente de distintos tipos de artefactos), un registro de modelos y activos con versionado (inspirado en MLflow pero extensible a scripts y dashboards), pruebas automatizadas incorporadas en el ciclo (siguiendo principios de DataOps de validar datos y modelos en cada cambio), y monitorización unificada con herramientas como Prometheus/Grafana para todas las piezas (datos, modelos, aplicaciones)

Al desplegarse mediante contenedores Docker e infraestructura como código (Terraform), DataOpsHub también se alinea con DevSecOps para ofrecer entornos reproducibles y seguros en la nube.

2.4. Tecnologías relacionadas con el tema del TFE

2.4.1. Angular (Frontend)

Angular es un framework web de código abierto mantenido por Google, basado en TypeScript, ideal para desarrollar aplicaciones de una sola página (SPA). Su arquitectura de componentes, tipado fuerte y herramientas integradas permiten construir interfaces modulares, seguras y fácilmente testeables. En DataOpsHub se emplea para ofrecer una interfaz de usuario dinámica y robusta, alineada con buenas prácticas de desarrollo frontend.

2.4.2. Python (Backend)

Python es un lenguaje de alto nivel ampliamente adoptado en desarrollo backend, análisis de datos y machine learning. Su claridad sintáctica y rico ecosistema de bibliotecas lo hacen ideal para construir APIs, ejecutar modelos ML y manipular artefactos analíticos. En DataOpsHub se utiliza como núcleo del backend, permitiendo una integración fluida entre la lógica de negocio, ejecución de procesos y servicios de datos.

2.4.3. Pytest y Unittest (Pruebas)

Pytest y unittest son frameworks de pruebas automatizadas en Python. Unittest, parte del estándar de Python, proporciona herramientas xUnit para pruebas unitarias básicas,

mientras que Pytest permite escribir pruebas más expresivas, parametrizadas y escalables. En conjunto, garantizan la validación continua del código de DataOpsHub, habilitando control de calidad durante el ciclo CI/CD.

2.4.4. Jenkins (CI/CD)

Jenkins es una plataforma de automatización ampliamente usada en DevOps para construir, probar y desplegar aplicaciones de manera continua. Su integración con múltiples herramientas y soporte para pipelines declarativos lo convierten en el núcleo del proceso CI/CD de DataOpsHub, gestionando desde la compilación hasta el despliegue automatizado en entornos cloud.

2.4.5. SonarQube (Calidad de Código)

SonarQube es una herramienta de análisis estático que evalúa calidad, seguridad y mantenibilidad del código fuente. Identifica errores, vulnerabilidades y code smells, generando métricas clave como cobertura de pruebas y deuda técnica. En DataOpsHub se usa como parte del pipeline Jenkins, asegurando que solo código confiable avance a producción.

2.4.6. Docker (Contenerización)

Docker permite empaquetar aplicaciones y sus dependencias en contenedores ligeros, garantizando que funcionen de forma idéntica en diferentes entornos. En DataOpsHub, se utiliza para contenerizar el backend, frontend y servicios auxiliares, facilitando despliegues consistentes, portables y escalables en la nube o entornos locales.

2.4.7. Terraform (Infraestructura como Código)

Terraform es una herramienta de infraestructura como código (IaC) que permite definir, versionar y desplegar entornos en la nube de forma declarativa. Gracias a su compatibilidad con múltiples proveedores, DataOpsHub puede aprovisionar recursos como contenedores, redes y almacenamiento bajo demanda, garantizando reproducibilidad y gobernanza en sus despliegues.

2.4.8. Prometheus, Loki y Grafana (Monitorización)

Prometheus recoge métricas en tiempo real, Loki almacena logs estructurados, y Grafana ofrece visualizaciones interactivas sobre ambos. Este stack de observabilidad brinda a DataOpsHub un sistema completo de monitoreo, alertas y trazabilidad, permitiendo detectar fallos, auditar procesos y optimizar el rendimiento operativo de la plataforma.

3. Objetivos y metodología de trabajo

3.1. Objetivo general

Diseñar e implementar una plataforma web unificada — DataOpsHub — que permita gestionar, auditar, ejecutar, versionar, monitorear y gobernar artefactos analíticos mediante principios de SecDevOps, utilizando tecnologías modernas como Angular, Python, Jenkins, Docker, Terraform, Prometheus, Grafana y SonarQube.

3.2. Objetivos específicos

- Crear una interfaz web en Angular que permita a los usuarios visualizar, cargar y ejecutar artefactos analíticos según permisos asignados.
- Desarrollar un backend en Python con autenticación, tokens, autorización por rol, auditoría y control de acceso a artefactos.
- Integrar pruebas automatizadas (unitarias, integración y E2E) usando Pytest, unittest y SonarQube.
- Diseñar procesos de reentrenamiento y ejecución de modelos ML, incluyendo auditoría y versionamiento.
- Orquestar jobs y automatizaciones desde la plataforma con monitoreo de su ejecución.
- Integrar monitoreo centralizado usando Prometheus, Grafana y Loki.
- Implementar CI/CD con Jenkins y Docker, asegurando calidad y despliegues controlados.

- Desplegar la plataforma en la nube con Terraform, habilitando portabilidad multi-cloud.
- Incorporar gobernanza y seguridad, permitiendo acceso diferenciado según rol, dominio de datos o área.

3.3. Metodología del trabajo

El proyecto seguirá un enfoque mixto basado en:

Metodología SecDevOps

- Integración temprana de seguridad en el ciclo de vida.
- Automatización de pruebas y auditoría.
- Contenerización del sistema para facilitar despliegue.
- Observabilidad como parte esencial del diseño.

Metodología ágil (Scrum)

El desarrollo se estructurará en tres sprints:

Sprint 1: Fundamentos y arquitectura

- Análisis de requisitos
- Diseño arquitectónico
- Creación del backend base
- Configuración de autenticación
- Construcción de repositorio Git
- Contenedorización inicial

Sprint 2: Plataforma funcional + CI/CD + pruebas

- Desarrollo de módulos del frontend
- Integración API + UI

- Implementación de pruebas unitarias y E2E
- Configuración de Jenkins
- Análisis SAST y SCA con SonarQube
- Versionamiento y ejecución de artefactos

Sprint 3: Monitoreo, gobernanza y despliegue cloud

- Integración con Prometheus, Grafana y Loki
- Orquestación de jobs
- Reentrenamiento de modelos
- Auditoría y logs estructurados
- Despliegue cloud con Terraform
- Validación y documentación final

4. Desarrollo específico de la contribución

En este bloque debes detallar el desarrollo de tu contribución. A continuación, te presentamos la estructura habitual para el tipo de trabajo considerado en este Máster. Tipo general: Desarrollo práctico. En este tipo de trabajo de desarrollo práctico, es importante justificar los criterios de planificación, análisis y diseño empleados para desarrollar el software, seguido de la descripción detallada del producto resultante y finalizando con una evaluación de la calidad y aplicabilidad del producto. Esto suele verse reflejado en la siguiente estructura de subapartados.

4.1. Planificación / Análisis / Requisitos

En este apartado se debe indicar el trabajo previo realizado para guiar el desarrollo del software. Esto debería incluir la identificación adecuada del problema a tratar, así como del contexto habitual de uso (empresa, institución, etc.), relacionado y complementando con lo que se haya podido incluir en los capítulos de introducción y contexto y estado del arte.

La tarea de planificación considerará todos los detalles a considerar para el desarrollo y el proceso. Idealmente, el análisis y la identificación de requisitos se debería hacer contando con expertos en la materia a tratar, en la medida de lo posible. Además, deberás describir en detalle las características del sistema. Como mínimo querrás mencionar:

- Qué tecnologías se utilizaron (incluyendo justificación de por qué se emplearon y descripciones detalladas de las mismas).
- Cómo se organizó el desarrollo.
- Qué personas participaron (con datos demográficos, si procede) o qué técnicas de sistemas se emplearon.
- Cómo transcurrió el desarrollo.
- Qué instrumentos de seguimiento y evaluación se utilizaron durante el proceso de desarrollo.

4.2. Descripción del sistema desarrollado / Implementación

Se deberán aportar detalles del proceso de desarrollo, incluyendo las fases e hitos del proceso. También deben presentarse diagramas explicativos de la arquitectura o funcionamiento, así como capturas de pantalla que permitan al lector entender el funcionamiento del programa. Igualmente, y muy importante, detallar las tecnologías que se han empleado (se puede hablar más sobre esto en el capítulo de contexto) y cómo se han empleado (sería lo principal a incluir en este capítulo de desarrollo), el uso de la metodología apropiada usada, así como las decisiones tomadas durante el proceso y la implementación llevada a cabo. Se pueden también incluir algunas cuestiones que se consideren interesantes para el entendimiento del desarrollo (cuando sean muy extensas se pueden llevar a los Anexos).

4.3. Evaluación

La evaluación debería cubrir al menos una mínima evaluación de la usabilidad de la herramienta, así como de su aplicabilidad para resolver el problema propuesto. Estas evaluaciones suelen realizarse con usuarios expertos, de ser posible, o con pruebas de usabilidad más básicas.

En cualquier caso, sería de especial relevancia y valor añadido el incluir también pruebas unitarias, de integración, según lo que corresponda, validando así la propuesta en la medida de lo posible. Esto es algo crucial para que queden más clarificadas las contribuciones que aporta el desarrollo.

A continuación, se indica cómo deben introducirse notas al pie, títulos y fuentes en Tablas y Figura, y Secciones.

EJEMPLO DE NOTA AL PIE

Ejemplo de nota al pie[1].

EJEMPLO DE TABLA

Tabla 1. ***Ejemplo de tabla con sus principales elementos.***

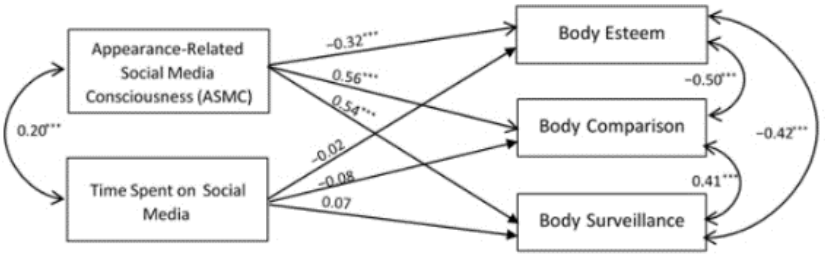
Measure	Urban		Rural		$F(1, 294)$	η^2
	M	*SD*	*M*	*SD*		
Self-esteem	2.91	0.49	3.35	0.35	68.87***	.19
Social support	4.22	1.50	5.56	1.20	62.60***	.17
Cognitive appraisals						
Threat	2.78	0.87	1.99	0.88	56.35***	.20
Challenge	2.48	0.88	2.83	1.20	7.87***	.03
Self-efficacy	2.65	0.79	3.53	0.92	56.35***	.16

Fuente: American Psychological Association, 2020e.

[1] Ejemplo de nota al pie.

EJEMPLO DE FIGURA

Figura 1. *Ejemplo de figura realizada para nuestro trabajo.*



Fuente: American Psychological Association, 2020f.

SECCIONES

4.3.1. “Título 3” del menú de estilos

4.3.2. “Título 3” del menú de estilos

4.3.2.1. "Título 4" del menú de estilos

4.3.2.2. "Título 4" del menú de estilos

5. Conclusiones y trabajo futuro

Este último bloque es habitual en todos los tipos de trabajos y presenta el resumen final de tu trabajo y debe servir para informar del alcance y relevancia de tu aportación.

5.1. Conclusiones

Suele estructurarse empezando con un resumen del problema tratado, de cómo se ha abordado y de por qué la solución sería válida.

Es recomendable que incluya también un **resumen de las contribuciones del trabajo**, en el que relaciones las contribuciones y los resultados obtenidos con los objetivos que habías planteado para el trabajo, discutiendo hasta qué punto has conseguido resolver los objetivos planteados. Las conclusiones ofrecidas deberán ser consecuencia del trabajo realizado y, por lo tanto, deberán marcar el grado de consecución de los objetivos propuestos (cada objetivo del trabajo se enlazará con una conclusión).

5.2. Líneas de trabajo futuro

Finalmente, se suele dedicar una última sección a hablar de **líneas de trabajo futuro** que podrían aportar valor añadido al TFE realizado. La sección debería señalar las perspectivas de futuro que abre el trabajo desarrollado para el campo de estudio definido. En el fondo, debes justificar de qué modo puede emplearse la aportación que has desarrollado y en qué campos.

Es importante tener presente que este apartado cierra el tema, no es un resumen del trabajo y no debe incluir datos nuevos.

Referencias bibliográficas

Una vez que el trabajo está terminado, hay que revisar el apartado de Referencias bibliográficas. Si has usado un sistema automático (un gestor bibliográfico tipo Endnote, Refworks o Mendeley), inserta la **bibliografía en la opción adecuada (APA)**.

Si lo has ido haciendo manualmente, repasa que todo es correcto: aparecen todas las referencias citadas en el texto, los autores están ordenados alfabéticamente por apellidos, las cursivas son correctas, los artículos tienen números de páginas, no faltan años ni ciudades de edición, se cumple en todas las referencias la normativa APA, etc.

Se recomienda evitar citas que hagan referencia a Wikipedia y que no todas las referencias sean solo enlaces de Internet, es decir, que se vea alguna variabilidad entre libros, congresos, artículos, proyectos, aplicaciones, herramientas, y enlaces puntuales de Internet.

EJEMPLO DE REFERENCIA BIBLIOGRÁFICA

Swanson, E., Barnes, M., Fall, A. M., & Roberts, G. (2017). Predictors of Reading Comprehension Among Struggling Readers Who Exhibit Differing Levels of Inattention and Hyperactivity. *Reading & Writing Quarterly, 34*(2), 132-146. doi:10.1080/10573569.2017.1359712

Anexo A. Título Anexo

En los anexos se recoge todo aquello que puede ser interesante para el trabajo pero que no es estrictamente esencial, y que distraería la lectura si se colocara en el cuerpo: cuestionarios, encuestas, código, resultados de pilotos, documentos adicionales, capturas de pantalla extra, etc. Pueden incluirse los anexos que se consideren necesarios. Estos no computarán a efectos de extensión del trabajo. Cada parte adicional se numera como un anexo y se presenta en una página diferente.