ESCUELA POLITÉCNICA DE INGENIERÍA DE GIJÓN

GRADO EN INGENIERÍA INFORMÁTICA EN TECNOLOGÍAS DE LA INFORMACIÓN

Lenguajes y Sistemas Informáticos

TRABAJO FIN DE GRADO/MÁSTER Nº ???

Explotación, integración y visualización de múltiples fuentes de datos mediante un Data Lakehouse

Mier Montoto, Juan Francisco

TUTORES:
D. Augusto Alonso, Cristian
D. Morán Barbón, Jesús
D. Vázquez Faes, Eduardo

FECHA: junio 2024

Índice de contenido

Índice de figuras

1. Introducción

En este capítulo se presenta una introducción al trabajo de desarrollo de software realizado, proporcionando un contexto general y estableciendo el escenario para los capítulos siguientes. Se discutirán los antecedentes y la motivación detrás de este trabajo, la finalidad del proyecto, y se proporcionará una breve descripción de la empresa en la que se ha desarrollado este trabajo. Este capítulo tiene como objetivo proporcionar una visión general del proyecto y establecer las bases para los capítulos detallados que siguen.

1.1. Antecedentes

Hoy en día, nos encontramos en una era donde la generación y almacenamiento de datos crece exponencialmente[1], reflejando una realidad ineludible en el ámbito empresarial. La diversidad de fuentes y formatos de estos datos introduce una complejidad significativa en su manejo, conocida como *heterogeneidad* [2], siendo las bases de datos, archivos de registros y APIs las fuentes más habituales. El término *big data* describe este fenómeno de acumulación masiva de datos, cuya magnitud y complejidad sobrepasan las capacidades de los métodos de procesamiento convencionales. Se distingue por tres características principales: volumen, variedad y velocidad. Su adecuada gestión y análisis pueden otorgar ventajas competitivas significativas a las empresas, tales como el descubrimiento de patrones ocultos, identificación de nuevas oportunidades de mercado y optimización de procesos de toma de decisiones.

Uno de los procesos que permite la extracción de esta información es la pirámide DIKW, [1] es un modelo que describe la relación entre los datos, la información, el conocimiento y la sabiduría. Según este modelo, los datos son la materia prima de la información, que a su vez es la materia prima del conocimiento, que a su vez es la materia prima de la sabiduría.

Una organización sin los procesos adecuados para la gestión y análisis de estos datos, se enfrenta a importantes desafíos, como la dificultad para identificar patrones y tendencias, la toma de decisiones incorrectas y la pérdida de oportunidades de negocio. Por otro lado,una organización que logre extraer información valiosa de sus datos, podrá mejorar su eficiencia, aumentar su competitividad y adaptarse mejor a un entorno empresarial en constante cambio.

La evolución tecnológica ha propiciado el desarrollo de innovadoras herramientas y metodologías diseñadas para enfrentar estos desafíos. Entre ellas, los *data lakes* (o *lagos de*

[1] https://www.statista.com/statistics/871513/worldwide-data-created/
[2] https://www.sciencedirect.com/topics/computer-science/data-heterogeneity

información) se destacan por su capacidad para consolidar vastos volúmenes de datos heterogéneos, facilitando su posterior análisis y aprovechamiento de manera más efectiva.

Sin embargo, el proceso de integración, visualización y análisis de estos datos es una tarea desafiante, ya que requiere de una gran cantidad de recursos y de un tiempo de desarrollo considerable del que, normalmente, no se dispone en el ámbito empresarial.

Con la ingesta masiva de datos, se presentan nuevos problemas a la hora de analizar y obtener información de ellos:

- Sin la necesaria automatización y correcta aplicación de los procesos ETL, al tratarse de un crecimiento exponencial de los datos y, por lo tanto, de la fuerza de trabajo necesaria para manejarla, el proceso de análisis puede dar lugar a errores y decisiones de negocio incorrectas además de un aumento de los costes.
- La heterogeneidad de los datos, tanto en formato como en origen, dificulta su consolidación y análisis.
- La masificación de información a representar requiere de herramientas de visualización, como *dashboards*, que permitan a su análisis.

La visualización de datos es una técnica que permite representar la información de manera visual, para facilitar su análisis y comprensión. La visualización de datos es una parte importante del proceso de análisis de datos, ya que permite identificar patrones, tendencias y anomalías en los datos de forma más rápida y sencilla.

1.2. Motivación

Okticket, como el resto de empresas, se enfrenta a la necesidad de gestionar y analizar grandes volúmenes de datos, provenientes de múltiples fuentes y en diferentes formatos. La correcta gestión y análisis de estos datos es fundamental para la toma de decisiones y para la mejora de los procesos internos de la empresa.

El proyecto surge de la necesidad de la empresa de extraer información y conocimiento de las múltiples y heterogéneas fuentes de datos de las que se disponen, tanto internas (e.g. bases de datos, archivos de registros, APIs, entre otros), como externas (e.g. APIs o datos de webs de terceros, datos de fuentes públicas. . .).

En la actualidad, la empresa dispone de una gran cantidad de datos que se encuentran en diferentes formatos y en diferentes ubicaciones, lo que dificulta su análisis y explotación. Por otra parte, se depende de la consulta manual o de servicios de terceros para poder analizar estos datos, lo que supone un coste adicional.

Además del uso interno, la empresa también quiere ofrecer a sus clientes la posibilidad de consultar estos datos de forma visual y sencilla, para que puedan analizarlos y explotarlos de forma autónoma, lo que supondría un valor añadido para los mismos.

1.3. Finalidad del proyecto

El objetivo de este proyecto es la creación de una herramienta que centralice y unifique las fuentes de datos heterogéneas. Esta consulta se realiza actualmente de manera manual mediante la creación y despliegue de un data lake (y todas las herramientas asociadas necesarias), con la finalidad de analizar los datos de forma más eficiente. El cumplimiento de este objetivo permitirá a la empresa obtener una serie de beneficios;

- **Eliminar el tiempo invertido** en la consulta manual de los datos.
- **Reducir los costes** asociados a servicios de terceros.
- **Mejorar la toma de decisiones**, al poder analizar los datos de forma más eficiente.
- **Incrementar la calidad de los servicios** ofrecidos a los clientes, al poder ofrecerles la posibilidad de consultar los datos de forma visual y sencilla.
- **Explotar económicamente** este servicio, ofreciéndolo a terceros.

Además de la mejora de los procesos ya existentes, la explotación mediante esta herramienta abrirá la puerta a nuevas posibilidades de análisis y explotación de los datos, como la detección de anomalías en la infraestructura o la predicción de patrones y eventos futuros.

1.3.1. Requisitos

Los anteriores objetivos se detallan en requisitos funcionales y no funcionales, que se presentan a continuación.

1.3.1.1. Requisitos funcionales

RF1. **Integración de fuentes de datos:** El data lake debe ser capaz de integrar datos de diferentes fuentes, de manera que se almacenen de manera unificada.

- RF1.1. *Fuentes internas*
  - RF1.1.1. MongoDB
  - RF1.1.2. MySQL
  - RF1.1.3. Okticket API
- RF1.2. *Fuentes externas*
  - RF1.2.1. APIs de terceros
  - RF1.2.2. Webs de terceros (*scraping*)

RF2. **Transformación de datos:** El data lake debe ser capaz de transformar los datos integrados en un formato que pueda ser analizado.

RF3. **Limpieza de datos:** El data lake debe ser capaz de limpiar los datos, para que estén preparados para el análisis.

- RF3.1. Eliminación de duplicados
- RF3.2. Eliminación de valores nulos
- RF3.3. Eliminación de valores atípicos
- RF3.4. Transformación de tipos de datos
- RF3.5. Transformación de fechas

RF4. **Gestón de metadatos:** El sistema debe gestionar metadatos, permitiendo a los usuarios entender el origen, el contenido y el contexto de los datos. A la hora de visualizar los datos, deben ser mostrados en la interfaz de consulta.

- RF4.1. Fuente de origen

RF4.2. Empresa vinculada

RF4.3. Fecha de creación

RF4.4. Fecha de actualización

RF5. **Soporte para análisis:** Debe permitir el análisis de datos complejos, incluyendo el procesamiento de grandes volúmenes de datos mediante búsquedas con el objetivo de segmentar la información.

RF6. **Interfaz de consulta:** Debe ser capaz de visualizar los datos de forma sencilla y eficiente, para que los usuarios puedan analizarlos y explotarlos de forma autónoma.

RF6.1. *Interfaz interna*

RF6.1.1. Monitorización de infraestructura

RF6.1.2. Análisis de negocio

RF6.1.3. Gestión de soporte

RF6.2. *Intefaz externa:* Consulta de datos a nivel externo para los administradores de empresas cliente que ofrezca un análisis a nivel de negocio de la información que se posea relacionada con dicha empresa.

RF7. **Gestión de calidad:** Herramientas y procesos para monitorear y mejorar continuamente la calidad de los datos almacenados en el datalake. Debe soportar la generación de informes de calidad de datos y la configuración de alertas basadas en umbrales de calidad de datos.

1.3.1.2. Requisitos no funcionales

RNF1. **Automatización**

RNF1.1. Los procesos ETL que alimentan el *data lake* deben ser automáticos.

RNF1.2. Los procesos de limpieza y transformación de datos deben ser automáticos.

RNF1.3. Los procesos de gestión de metadatos deben ser automáticos.

RNF1.4. El proceso de despliegue del programa debe estar automatizado (pipelines, docker/kubernetes, etc.)

RNF2. **Escalabilidad**

RNF2.1. El sistema debe ser escalable horizontalmente, para que pueda soportar el crecimiento de los datos.

RNF2.2. El sistema debe ser escalable verticalmente, para que pueda soportar el crecimiento de los recursos de procesamiento.

RNF3. **Disponibilidad**

RNF3.1. El sistema debe ser disponible 24/7, para que pueda ser utilizado por los usuarios en cualquier momento.

RNF3.2. El sistema debe contar con al menos 3 nueves de disponibilidad (99,9 %).

RNF4. **Seguridad de la información:** El sistema debe garantizar la protección de la información y asegurar que los datos no puedan ser accedidos por ningún agente externo.

RNF5. **Rendimiento**

RNF5.1. La latencia de navegación entre el usuario y el sistema debe ser inferior a 2 segundos.

RNF5.2. La latencia de consultas de datos debe ser inferior a 15 segundos.

RNF6. **Eficiencia de costes:** El sistema debe ser eficiente en el coste de desarrollo, tanto en el tiempo como en el coste de recursos.

RNF7. **Mantenibilidad:** el sistema y su código debe de ser mantenible e introducir buenas prácticas de manera que se facilite el desarrollo futuro sobre el mismo.

RNF8. **Cumplimiento normativo y privacidad:** El sistema debe cumplir con toda la legislación y normativa relativa a la privacidad de los datos, incluyendo la normativa de la empresa y las certificaciones a la que está sometida.

1.4. La empresa

Okticket es una startup nacida en Gijón en 2017 cuyo producto principal es un servicio software que escanea automáticamente de tickets y notas de gastos lo que permite reducir los costes y el tiempo que invierten las empresas en contabilizar y manejar los gastos de viaje de los profesionales.

La empresa tienen su suede principal en el Parque Tecnológico de Gijón, aunque cuenta con un número de sedes creciente en varios países, como Francia, Portugal o, más recientemente, México. En esta oficina principal se encuentran los departamentos de ventas y marketing, así como el equipo de desarrollo y soporte.

Okticket es una de las empresas que más crecen tanto del sector como del propio Parque Tecnológico. Debido a este rápido crecimiento, el equipo está en constante desarrollo y cambio, tanto aquí en España como en el resto de sedes. Este crecimiento se refleja en la recepción de un gran número de galardones y reconocimientos. [3] [4] [5] [6]

La parte principal del negocio es el núcleo del software como servicio (Software as a Service en inglés, en adelante *SaaS*), es decir, la aplicación completa tanto para administradores como para empleados. Este SaaS se oferta a empresas de cualquier tamaño, cuyo precio final varía en función del número de usuarios, las características e integraciones que requiera la empresa cliente y el soporte que se ofrezca.

Recientemente se han añadido nuevas propuestas a la cartera de servicios ofertada por Okticket, como la OKTCard - una tarjeta inteligente que gestiona automáticamente los gastos, así como la inclusión de nuevos “módulos” de gestión de gastos y viajes.

Debido a todo este crecimiento, la empresa maneja una gran cantidad de datos importantes que se encuentran en diferentes formatos y en diferentes ubicaciones, lo que dificulta su análisis y explotación. Por otra parte, se depende de la consulta manual o de servicios de terceros para poder analizar estos datos, lo que supone un coste adicional.

[3] Okticket en el especial startups 2023 de Forbes (LinkedIn)
[4] Arcelor y Okticket, premios nacional de Ingeniería Informática (EL COMERCIO)
[5] Okticket recibe el sello Pyme Innovadora (okticket.es)
[6] Okticket, empresa emergente certificada (okticket.es)

2. Fundamento teórico

En este capítulo se presentan los conceptos y términos fundamentales que se utilizan en el proyecto, para proporcionar una base teórica sobre la que se desarrolla el trabajo. Se discuten los conceptos de *data lake*, *procesos ETL* y *dashboards*, que son fundamentales para el desarrollo del proyecto.

2.1. Paradigmas de almacenamiento de datos

2.1.1. Data warehouse

Un *data warehouse*[1], también conocido en español como almacén de datos, es una base de datos que se utiliza para almacenar y analizar grandes cantidades de datos de manera eficiente. Los almacenes de datos proporcionan acceso rápido y compatible con plataformas de consultas (como SQL) a grandes cantidades de datos, lo que permite a los analistas y a los científicos de datos realizar análisis complejos sobre los datos almacenados.

Todos los datos almacenados en un *data warehouse* se encuentran en un formato común, para lo que se aplican procesos ETL (extracción, transformación y carga) que transforman los datos de diferentes fuentes en un formato común. Esto significa que la información se encuentra en un formato o esquema optimizado y específico, lo que facilita su manipulación y análisis pero limita la flexibilidad al acceso de los datos y genera costes adicionales en el caso de tener que modificar o transferir los mismos para su uso.

2.1.2. Data lake

Los *data lakes*[2] son almacenes de datos que guardan grandes cantidades de datos de manera no estructurada [2]. En el ámbito de una empresa, un *data lake* contiene datos de diferentes fuentes de valor no considerado hasta su análisis, de manera que su explotación posterior y su análisis no depende de una estructuración y transformación compleja, reduciendo los costes de los procesos ETL derivados, una flujo de tareas que se aplican sobre la información para ingestarla. Esto no quiere decir que no se apliquen estos procesos a los datos, sino que se aplican de manera más flexible y básica que en otras estructuras de almacenamiento de datos con esquemas predefinidos, como los *data warehouses*. [3]

[1] `https://aws.amazon.com/es/data-warehouse/`
[2] `https://aws.amazon.com/es/what-is/data-lake/`

A diferencia de los *data warehouses*, los *data lakes* no tienen un esquema definido, lo que permite almacenar datos *heterogéneos*. Esto permite almacenar grandes cantidades de información sin tener que definir un esquema de antemano, lo que puede ser útil en aquellos casos en los que no se conoce la estructura de los datos que se van a almacenar.

Estas características de los *data lakes* hacen que sean más atractivos en el sector empresarial de cara al análisis de información, en contraste con las estructuras planteadas normalmente en el campo de la investigación académica.

Para consultar esta gran cantidad de datos almacenados, se suelen utilizar técnicas de visualización de datos, como los *dashboards*, herramientas de visualización que permiten observar los datos de manera sencilla y eficiente.

2.1.3. Data lakehouse

Los *data lakehouses* son una combinación funcional de los dos paradigmas vistos anteriormente, los *data lakes* y los *data warehouses*. Los *data lakehouses* permiten almacenar datos tanto de manera estructurada como no estructurada, lo que facilita aprovechar la información al contar con una única estructura de bajo coste que ofrece a los usuarios que lo necesiten explorar y analizar los datos según sus necesidades.

Puesto que en este proyecto se plantea el uso de datos tanto estructurados como no estructurados, y no siempre será de interés aplicar procesos de transformación a toda la información obtenida, los *data lakehouse* se presentan como una solución eficiente y flexible para el almacenamiento y análisis de los datos.

2.2. Procesos ETL

Los procesos ETL [2] son procesos que combinan datos de múltiples fuentes en un único destino, transformando los datos en un formato común. Estos procesos se utilizan para extraer datos de diferentes fuentes, transformarlos en un formato común y cargarlos en un destino común, como puede ser un *data lake*.

Los procesos ETL, fundamentales en el ámbito de la gestión de datos, presentan atributos distintivos que facilitan la integración eficaz de información procedente de diversas fuentes:

- **Adaptabilidad:** los procesos ETL deben de adaptarse a la estructura de los datos de la fuente de origen, ya que dichas fuentes pueden tener diferentes estructuras y tener tipos de datos diferentes (la característica de *heterogeneidad* de los datos que ya se ha mencionado).
- **Escalabilidad:** otra de las características clave de los procesos ETL es que sean escalables, ya que los datos que se muestran en los dashboards suelen ser datos que se generan de manera continua, y por lo tanto los procesos ETL deben ser capaces de procesar grandes cantidades de datos de manera eficiente. En ocasiones, los procesos ETL se pueden realizar en *streaming*, lo que significa que los datos se procesan en tiempo real a medida que se generan.

2.2.1. Funcionamiento

Los procesos ETL se dividen en tres fases principales: *extracción*, *transformación* y *carga*, como se muestra en el sigiuente diagrama:

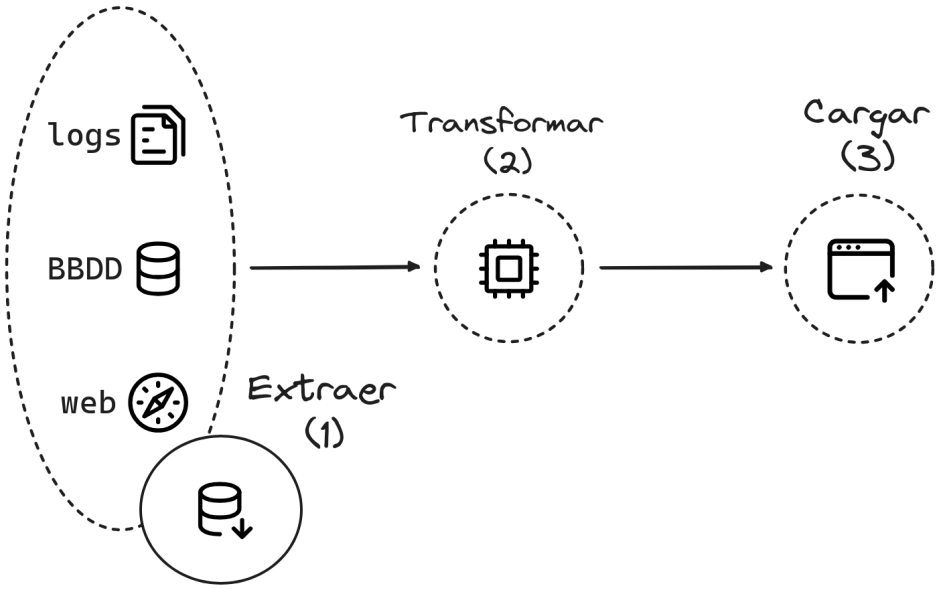


Figura 2.1: Fases de un proceso ETL (trabajo propio)

Al comienzo del proceso, se tienen datos presúntamente heterogéneos que no se pueden analizar de manera eficiente. Tras aplicar todos los pasos de las fases anteriores, se obtiene un conjunto de datos homogéneos y listos para ser analizados en el destino indicado (en el caso de este proyecto, un *data lake*).

Extracción (1) En este proceso se extraen los datos de las fuentes de datos, que pueden ser bases de datos, logs, APIs, etc. En este paso, se pueden aplicar filtros para extraer solo los datos que se necesiten, y se pueden extraer datos de múltiples fuentes *heterogéneas*.

En algunos casos, los datos se pueden extraer de manera incremental, es decir, solo se extraen los datos que han cambiado desde la última extracción. Esto puede ser útil para reducir el tiempo de procesamiento y el volumen de datos que se almacenan.

En otros casos, los datos se pueden extraer de manera continua, es decir, se extraen los datos en tiempo real según se van generando. Esto puede ser útil para procesar datos que se generan en tiempo real, como logs o datos de sensores.

Transformación (2) En este proceso se transforman los datos extraídos en la fase anterior, normalmente aplicándoles un proceso de limpieza y transformación a un formato común. En este paso, se pueden aplicar diferentes operaciones a los datos, como la limpieza, la agregación, la normalización, la conversión de formatos, etc.

Una transformación especialmente importante que se suele reaizar durante este proceso es la limpieza, revisión y corrección de los datos extraídos, para asegurar que se almacena información correcta y consistentes. Durante esta fase se contemplan operaciones más complejas, como pueden ser la agregación de datos, la conversión de formatos, la normalización de datos, el cifrado, etc.

Estos procesos de transformación son vitales cuando el sistema maneja una gran cantidad de datos heterogéneos de múltiples fuentes de manera simultánea, como puede ser el caso de un *data lake* o un *data warehouse*. En el caso del primero, no es necesaria la transformación de los datos a un formato común, pero si otros procesos clave como la limpieza y la normalización de los datos, entre otros.

Carga (3) En este proceso se cargan los datos transformados en el destino final. Frecuentemente, los datos se almacenan en una data warehouse o data lake para su posterior análisis.

La frecuencia del proceso de carga depende de la naturaleza de los datos y de las necesidades del negocio, como ya se ha descrito en el proceso de extracción.

Tras completar este proceso, los datos están listos para ser analizados y visualizados desde las arquitecturas de datos que almacenen la información.

2.2.2. Alternativas

Aunque lo más común es el flujo anteriormente explicado de *extracción*, *transformación* y *carga*, existen algunos flujos y procesos alternativos que evitan algunos de estos pasos, normalmente en casos específicos que se beneficien del cambio:

- **Virtualización de datos:** capa virtual de abstracción que permite acceder a los datos de las fuentes sin necesidad de extraerlos. Esto permite ahorrar espacio de almacenamiento y tiempo de procesamiento, pero suele ser menos eficiente en términos de rendimiento y no es compatible con todas las arquitecturas de datos.

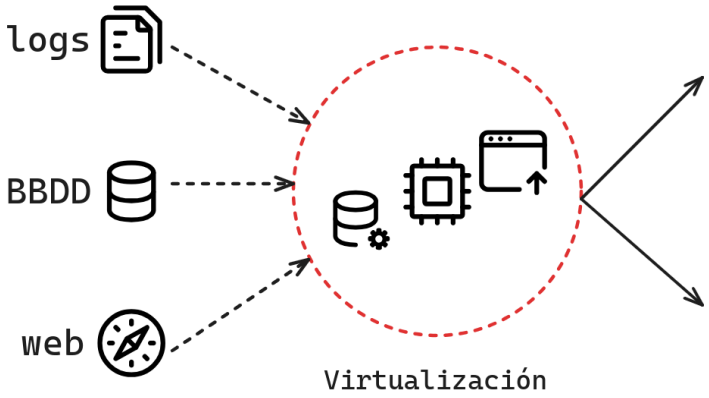


Figura 2.2: Ejemplo de flujo con virtualización (trabajo propio)

- **Proceso *ELT*[3]:** en lugar de transformar los datos antes de cargarlos en el destino, se cargan los datos en bruto y se transforman en el destino. Funciona bien para grandes conjuntos de datos sin estructura que requieran una carga (o recarga) contínua, aunque, al igual que la virtualización, puede ser menos eficiente o incompatible con algunas arquitecturas de datos, como los *data warehouses*.

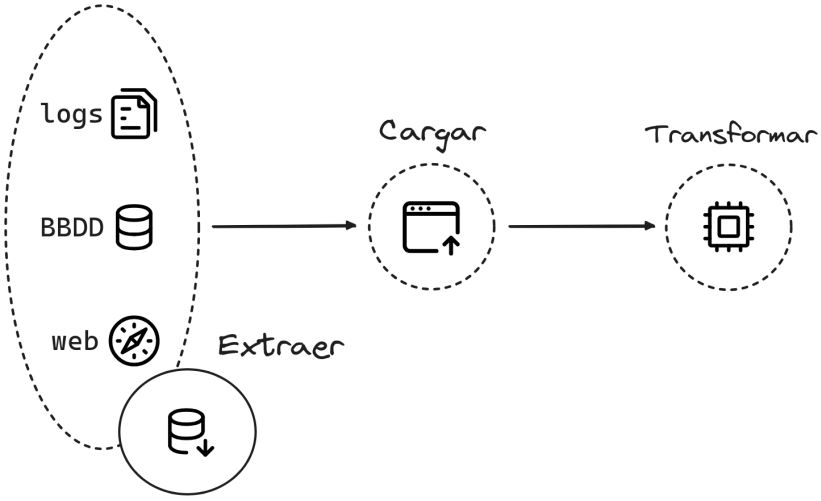


Figura 2.3: Diagrama de flujo de un proceso *ELT* (trabajo propio)

[3] https://www.ibm.com/topics/elt

2.3. Dashboards

Definición La palabra *dashboard*, que traducido de manera literal significa *cuadro de mandos*, es un término que se utiliza para referirse a cualquier interfaz gráfica que muestre información relevante de manera visual sobre un proceso o negocio. Aunque el término se utiliza en muchos ámbitos: indicadores comerciales, de producción, de marketing, de calidad, de recursos humanos…, en este proyecto se utilizará en el ámbito de la monitorización de sistemas y procesos de negocio.

En el ámbito de deste proyecto, los dashboards reflejan en tiempo real el rendimiento de actividades o procesos de negocio, y se utilizan para tomar decisiones informadas sobre los mismos. En el caso de una empresa, un dashboard puede mostrar desde el rendimiento de la plataforma en tiempo real hasta un reflejo del de las ventas, y permitir a los directivos tomar decisiones informadas sobre el futuro de la empresa.

Características Los dashboards cuentan con una serie de características que los hacen útiles para la toma de decisiones: [2]

- **Visualización de datos:** es la característica fundamental de cualquier cuadro de mandos, y aquella que determina su utilidad. La visualización de datos es la ciencia de presentar los datos de manera que se pueda extraer información útil y realizar decisiones informadas sobre ellos. Un buen dashboard cuenta con gráficas, tablas, indicadores, etc. que permiten al usuario entender la información que se está presentando con un conocimiento técnico mínimo.
- **Interactividad y personalización:** un dashboard debe permitir al usuario interactuar con los datos (filtrarlos, ordenarlos, profundizar en ellos...) y ajustar la información que se muestra a cada proceso o negocio que se esté evaluando. Esta capacidad asegura que el dashboard se adapte tanto a las necesidades actuales como a las evoluciones futuras de lo que se esté analizando.
- **Accesibilidad y portabilidad:** un dashboard debe ser accesible desde una variedad de situaciones y dispositivos, manteniendo su funcionalidad y forma. Aunque normalmente los dashboards se analizan en pantallas grandes, es importante que también se puedan consultar en otras circunstancias, como dispositivos móviles.

2.3.1. Dashboards planteados

Para el sistema que se describe, se plantean dos tipos de dashboards diferentes:

- **Dashboards internos**: que reflejan el rendimiento de la plataforma en tiempo real.
- **Dashboards externos**: que reflejan el rendimiento de las ventas y permiten a los clientes tomar decisiones informadas sobre su negocio.

Hay que destacar que los dashboards externos son diferentes a los dashboards de monitorización, tanto en el contenido que presentan como en la manera de acceder a ellos: mientras que los dashboards de monitorización serán de uso interno exclusivo, los dashboards externos deben estar disponibles para los clientes de las empresas que los contraten.

3. Planificación del proyecto

La planificación de un proyecto es fundamental para su correcto funcionamiento y desarrollo, dentro de los plazos y costes establecidos. En este apartado se detallan las tareas que se llevarán a cabo en el proyecto, así como los recursos necesarios y los plazos de ejecución.

3.1. Metodología

En este capítulo se aborda la metodología adoptada para el desarrollo del proyecto, fundamentada en principios ágiles y enfocada en la entrega continua de valor. La elección de *Scrum*, una metodología que permite elaborar productos software de manera incremental, revisando el producto continuamente y adaptándolo a las necesidades del cliente, como marco de trabajo subraya nuestro compromiso con la adaptabilidad y la mejora continua.

La estructura de este capítulo se organiza en torno a la descripción detallada de la metodología *Scrum*, la visualización de la planificación y las estrategias de comunicación adoptadas. A través de esta metodología, buscamos optimizar los recursos disponibles, ajustarnos a los plazos establecidos y garantizar la calidad del producto final.

La implementación de *Scrum* se complementa con herramientas de visualización y gestión de proyectos, como los tableros de *GitHub*, que facilitan la organización y seguimiento de las tareas. Además, se pone especial énfasis en la comunicación efectiva dentro del equipo de desarrollo y con los stakeholders, asegurando así una alineación constante con los objetivos del proyecto.

Este enfoque metodológico no solo refleja la planificación y ejecución del proyecto, sino que también establece las bases para una gestión eficaz, adaptativa y orientada a resultados.

3.1.1. Scrum

Para la planificación del proyecto se ha escogido *Scrum*, una metodología "ágil" que se basa en la realización de iteraciones cortas y en la adaptación a los cambios. La metodología *Scrum* se estructura en *sprints* (iteraciones cortas de una duración fija), en las que se llevan a cabo una serie de tareas que se han planificado previamente.

El primer paso de la metodología *Scrum* es la creación de un *product backlog*, una lista ordenada de las tareas a realizar durante el desarrollo del producto, a partir de los requisitos

del sistema, que a su vez son una versión refinada de los requisitos iniciales del proyecto. A partir de este *product backlog* se planifican las tareas que se llevarán a cabo en cada *sprint*, de manera que sea posible cumplir con los objetivos del proyecto en el tiempo establecido.

A diferencia de metodologías tradicionales o *en cascada*, *Scrum* permite la adaptación a los cambios y la mejora continua del producto, ya que se revisa y se adapta en cada *sprint* según las necesidades del cliente y del equipo de desarrollo. Por otro lado, *Scrum* se diferencia de otras metodologías más ágiles como *XP* en que no se centra tanto en las prácticas de desarrollo, sino en la gestión del proyecto y en la entrega de valor al cliente.

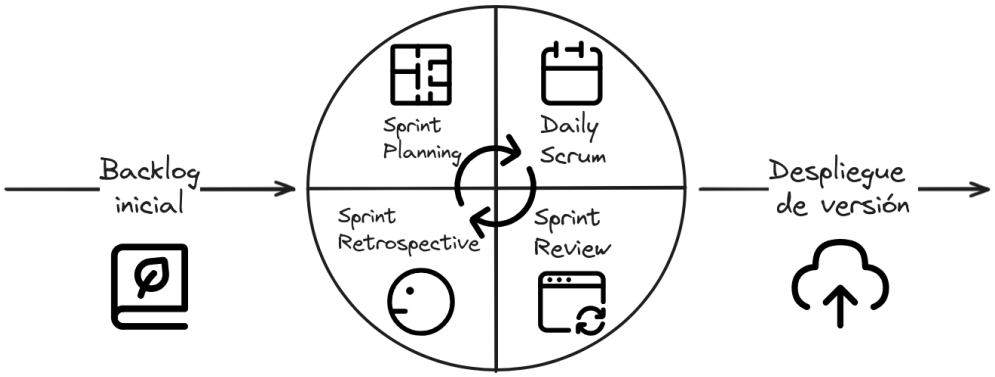


Figura 3.1: Diagrama de la metodología *Scrum*

Roles En *Scrum* se distinguen tres roles principales:

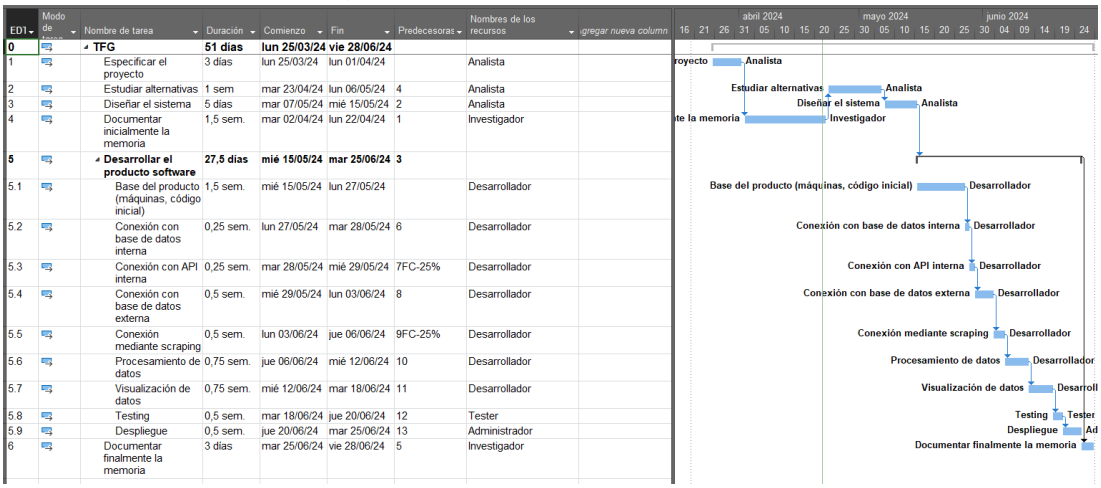
- **Product Owner**: es la persona responsable de definir los requisitos del producto y de priorizar las tareas del *product backlog*. Es el enlace entre el equipo de desarrollo y el cliente, y es el responsable de garantizar que el producto cumple con las expectativas del cliente.
- **Scrum Master**: es la persona responsable de garantizar que el equipo de desarrollo sigue la metodología *Scrum* y de eliminar los obstáculos que puedan surgir durante el desarrollo del proyecto. El *Scrum Master* es el encargado de organizar las reuniones diarias y de asegurar que el equipo de desarrollo cumple con los plazos y los objetivos del proyecto.
- **Equipo de desarrollo**: es el equipo encargado de llevar a cabo las tareas del *product backlog* y de entregar el producto final. El equipo de desarrollo es autoorganizado y multidisciplinario, y se organiza en torno a las tareas que se van a realizar en cada *sprint*.

3.1.2. Visualización de la planificación

Para la visualización de la planificación se ha utilizado la herramienta de gestión de proyectos de *GitHub*, que permite múltiples visualizaciones de tareas e *issues* en tableros separados.

- Se utiliza un tablero de *requisitos* al estilo *Kanban* para visualizar los requisitos del proyecto y su estado, siguiendo con la metodología *Scrum*. Un tablero *Kanban* es una herramienta visual que permite gestionar el flujo de de trabajo de un proyecto por "sprints", dividiendo las tareas en columnas y moviéndolas de una columna a otra según su estado.

- Se utiliza un diagrama Gantt para visualizar las tareas y los plazos de ejecución de cada una de ellas, de manera que sea posible llevar un control de los plazos y de las tareas que se están realizando en cada momento, además de facilitar la creación del presupuesto inicial.



EDT	Modo de tarea	Nombre de tarea	Duración	Comienzo	Fin	Predecesoras	Nombres de los recursos
0		**TFG**	**51 días**	**lun 25/03/24**	**vie 28/06/24**		
1		Especificar el proyecto	3 días	lun 25/03/24	lun 01/04/24		Analista
2		Estudiar alternativas	1 sem	mar 23/04/24	lun 06/05/24	4	Analista
3		Diseñar el sistema	5 días	mar 07/05/24	mié 15/05/24	2	Analista
4		Documentar inicialmente la memoria	1,5 sem.	mar 02/04/24	lun 22/04/24	1	Investigador
5		**Desarrollar el producto software**	**27,5 días**	**mié 15/05/24**	**mar 25/06/24**	**3**	
5.1		Base del producto (máquinas, código inicial)	1,5 sem.	mié 15/05/24	lun 27/05/24		Desarrollador
5.2		Conexión con base de datos interna	0,25 sem.	lun 27/05/24	mar 28/05/24	6	Desarrollador
5.3		Conexión con API interna	0,25 sem.	mar 28/05/24	mié 29/05/24	7FC-25%	Desarrollador
5.4		Conexión con base de datos externa	0,5 sem.	mié 29/05/24	lun 03/06/24	8	Desarrollador
5.5		Conexión mediante scraping	0,5 sem.	lun 03/06/24	jue 06/06/24	9FC-25%	Desarrollador
5.6		Procesamiento de datos	0,75 sem.	jue 06/06/24	mié 12/06/24	10	Desarrollador
5.7		Visualización de datos	0,75 sem.	mié 12/06/24	mar 18/06/24	11	Desarrollador
5.8		Testing	0,5 sem.	mar 18/06/24	jue 20/06/24	12	Tester
5.9		Despliegue	0,5 sem.	jue 20/06/24	mar 25/06/24	13	Administrador
6		Documentar finalmente la memoria	3 días	mar 25/06/24	vie 28/06/24	5	Investigador

Figura 3.2: Diagrama Gantt de tareas y plazos de ejecución

- Se utiliza un *roadmap* de apartados de la memoria, donde se visualiza su estado y sus fechas límite. Este *roadmap* no está relacionado con la metodología *Scrum*, sino que se ha creado a propósito para facilitar la visualización del progreso de cada sección y de la memoria en general.

Figura 3.3: Roadmap de apartados de la memoria

3.1.3. Comunicación

La comunicación con los tutores y con el equipo de desarrollo se considera fundamental para el correcto desarrollo del proyecto. Puesto que el trabajo se desarrolla de manera presencial en la oficina de la empresa, la comunicación con el equipo de desarrollo se realiza de manera frecuente y directa, mientras que la comunicación con los tutores se realiza de manera remota pero igual de frecuente, manteniendo el contacto mediante correo electrónico y Teams para pedir revisiones e informar sobre el estado del trabajo en todo momento.

3.1.4. Plataformas de planificación

Con el objetivo de facilitar las tareas de desarrollo y cumplimentar los requisitos por parte de la empresa, se utilizan las siguientes plataformas y herramientas de desarrollo para la fabricación del proyecto:

- **GitHub**: Plataforma de desarrollo colaborativo para el desarrollo del proyecto. Se utiliza para la gestión de tareas, seguimiento de desarrollo, documentación y colaboración.
- **Atlassian suite (*Jira, Bitbucket*)**: Suite de herramientas de gestión de proyectos y desarrollo colaborativo. Se utiliza para el desarrollo y documentación del proyecto de parte de la empresa.
- **Microsoft Teams**: Herramienta de comunicación y colaboración en tiempo real.
- **Microsoft Outlook**: Herramienta de comunicación por correo electrónico.

3.2. Presupuesto

4. Análisis

Este capítulo se centra en desglosar los componentes críticos del proyecto, específicamente dirigido a entender las necesidades de Okticket y cómo el desarrollo propuesto se alinea con estas. Se analizarán los requisitos funcionales y no funcionales, evaluando cómo cada uno contribuye al éxito del proyecto. Además, se identificarán las partes interesadas clave y se explorarán sus expectativas y requisitos, para asegurar que el sistema desarrollado cumpla con sus necesidades específicas. Este análisis detallado tiene como objetivo final proporcionar una hoja de ruta clara para el desarrollo del proyecto, asegurando que se tomen decisiones informadas que maximicen el valor entregado a la empresa y sus clientes.

4.1. Partes interesadas (stakeholders)

Las partes interesadas en el proyecto son aquellas personas o entidades que tienen un interés en el mismo, ya sea porque se ven afectadas por el resultado del proyecto, o porque tienen algún tipo de interés en el mismo. Las partes interesadas en este proyecto son las siguientes:

1. **Okticket**: la empresa es la principal parte interesada en el proyecto, ya que es la que se beneficiará directamente de los resultados del mismo, así como de las oportunidades de negocio que se abren con la explotación de los datos.
    - **Equipo de desarrollo de la empresa**: el equipo de desarrollo es otra parte interesada en el proyecto, ya que son los encargados de llevar a cabo la implementación del sistema y de garantizar su correcto funcionamiento, además de gestionar el soporte de servicio a nivel técnico.
    - **Equipo de soporte de la empresa**: el sistema planteado ahorraría tiempo al equipo de soporte, ya que les permitiría analizar los datos de forma más eficiente e identificar problemas antes de que tener que resolver las peticiones de los clientes afectados a nivel básico.
2. **Clientes**: los clientes de la empresa también son partes interesadas, puesto que se beneficiarán de los nuevos servicios que se ofrecen, como los dashboards de negocio que se han descrito anteriormente.
3. **Investigador y desarrollador (*Mier Montoto, Juan Francisco*):** el desarrollador del proyecto tiene la oportunidad de aplicar los conocimientos adquiridos en el desarrollo de un proyecto real, y de adquirir nuevos conocimientos en el proceso.

4.2. Valoración de alternativas

4.3. Definición del sistema

5. Diseño del sistema

5.1. Arquitectura del sistema

5.2. Modelo de datos

6. Implementación

7. Resultados

8. Conclusiones y trabajo futuro

Bibliografía

[1] Wikipedia contributors, "Dikw pyramid — Wikipedia, the free encyclopedia." https://en.wikipedia.org/w/index.php?title=DIKW_pyramid&oldid=1211227190, 2024. [Online; accessed 7-April-2024].

[2] J. Mier, "Presentación de datos: dashboards y procesos ETL." Primera entrega de teoría de la asignatura Inteligencia de Negocio, EPI Gijón, curso 23-24, 2023.

[3] P. Khine and Z. Wang, "Data lake: a new ideology in big data era," *ITM Web of Conferences*, vol. 17, p. 03025, 01 2018.

[4] J. Mier, "latexTemplate." https://github.com/miermontoto/latexTemplate, 2024. Plantilla de LaTeX personal para trabajos académicos.

[5] J. Mier, "Minería de anomalías." Segunda entrega de teoría de la asignatura Inteligencia de Negocio, EPI Gijón, curso 23-24, 2024.