Universidad de Valladolid

Escuela de Ingeniería Informática

Trabajo Fin de Grado

Grado en Ingeniería Informática

Automatización de procesos ETL para la gestión de acuerdos de servicio (SLA)

Autor:
Dña. Paula Pastor Gómez

Universidad de Valladolid

Escuela de Ingeniería Informática

Trabajo Fin de Grado

Grado en Ingeniería Informática

Automatización de procesos ETL para la gestión de acuerdos de servicio (SLA)

Autor:
Dña. Paula Pastor Gómez

Tutor:
D.Manuel Barrio Solorzano

Resumen

Hoy en día, el análisis de datos es un factor estratégico para cualquier empresa que desee alcanzar el éxito y sobresalir respecto a otras de su sector. En una sociedad cada vez más informatizada y en continuo cambio ya no es suficiente con hacer uso del sentido común a la hora de tomar las decisiones sobre un negocio, sino que se deben utilizar todas las herramientas e información que tenemos a nuestra disposición. Ya no sólo para mejorar el rendimiento sino para poder sobrevivir en un mercado cambiante.

Cualquier empresa, independientemente de su tamaño, recopila y almacena gran cantidad de datos que puede ser de gran interés para sus objetivos comerciales, no sólo para cumplirlos sino para superarlos continuamente. A medida que crece una empresa así lo hacen sus datos, pero la mayoría de las empresas no están aprovechando su potencial. Los datos apenas tienen valor por sí solos y no sólo sirve con recopilarlos, es necesario convertirlos en información útil que aporte conocimiento a la empresa. Mediante técnicas de Business Intelligence, una empresa será capaz de integrar, depurar, transformar y modelar sus datos para poder extraer información relevante que le permita descubrir y corregir errores sobre su organización y le ayude en la toma de decisiones.

Por ello, este proyecto pretende ser un acercamiento a la analítica de datos y a las tecnologías de business inteligence, que permitirá analizar el rendimiento de un negocio sobre una determinada área o interés, y le ayudará a respaldar las acciones y decisiones que tome en un futuro.

Abstract

Data analysis nowadays is a strategic factor for any company wishing to succeed and to stand out from its competitors. In an increasingly computerized and ever-changing society, it is no longer enough to use common sense when making decisions about a business; all the tools and information at our reach must be used. Not only to improve performance, but also to survive in an evolving market.

Any company, no matter its size, collects and stores large amounts of data that can be of great interest to its business goals, not only to meet them but also to continually exceed them. As a company grows so does its data, but most companies are not taking advantage of its potential. Data has little value on its own, and it is not enough just to collect it; it must be converted into useful information that provides knowledge to the company. By means of Business Intelligence techniques, a company will be able to integrate, clean, transform and model its data to extract relevant information that will allow it to discover and correct errors in its organization and help in the decision-making process.

Therefore, this project aims to be an approach to data analytics and business intelligence technologies, which will allow the analysis of the performance of a business on a certain domain or interest, and will help to support the actions and decisions to be adopted in the future.

Tabla de Contenidos

Lista de Figuras

Lista de Tablas

Capítulo 1

Introducción

1.1. Vocabulario Común y Acrónimos

Glosario de términos, conceptos y acrónimos ordenados alfabéticamente:

- **BBDD**: Bases de Datos
- **CRM**: siglas utilizadas para Customer Relationship Manager
- **Cron**: proceso que permite programar la ejecución de tareas de forma automática.
- **ERP**: siglas en ingles de "Planificación de Recursos Empresariales"
- **ETL**: siglas en ingles de Extract, Transform, Load (Extraer, Transformar, Cargar).
- **ITIL**. Biblioteca de Infraestructura de Tecnologías de Información
- **KPI**. Medidor de desempeño. Métrica que mire la eficacia y productividad.
- **SLA**. siglas en ingles de Service Level Agreement. Contrato que describe el nivel de servicio que un cliente espera de la empresa una empresa de servicios. Para este estudio serán servicios de telecomunicaciones.
- **TI**: Tecnología de la Información

1.2. Planteamiento del problema y Objetivos

Hoy en día las empresas almacenan en sus bases de datos gran cantidad de información de sus operaciones diarias, pero dicha información carece de valor por sí sola. Los datos son un activo muy valioso que permite a las empresas tomar decisiones y diseñar estrategias de negocio, pero para que puedan hacerlo de una manera efectiva deben ser capaces de gestionar estos datos correctamente, ya que no importa la cantidad sino la calidad de estos. Las empresas deben ser capaces de extraer, procesar, almacenar y analizar sus datos de negocio.

Este proyecto se va a desarrollar en el ámbito de una consultoría tecnológica que pretende dar soporte a la toma de decisiones de una empresa proveedora de servicios cuyo propósito es determinar si está cumpliendo unos SLAs. El cumplimiento de sus acuerdos SLA es una actividad esencial para cualquier empresa de TI, pero en ocasiones, debido a la cantidad de proyectos, cargas de trabajo y el manejo de

incidencias o emergencias, la realización de estos seguimientos puede verse afectado e incluso pasarse por alto. Sin embargo, es muy importante conocer cuándo y con qué frecuencia se están incumpliendo los plazos de estos acuerdos, para poder descubrir las causas de sus infracciones y realizar acciones al respecto. Existen otras razones por las cuales es importante medir el cumplimiento de estos contratos, como por ejemplo:

- Evitar malas experiencias con el cliente derivando en la pérdida de los mismos, lo cual repercutirá en los ingresos y la reputación de la empresa.
- Evitar las posibles indemnizaciones que puedan suponer el incumplimiento de estos contratos.
- Para ayudar al equipo encargado de resolver las incidencias a conocer y priorizar los problemas de los que debe hacerse cargo.
- Para identificar y resolver problemas de rendimiento.
- Para conocer las necesidades de personal y poder dotar de más personas si fuera necesario, etc.

El objetivo principal de este proyecto será establecer un marco tecnológico que permita realizar la automatización de procesos de cálculo de SLAs, desde la ingesta de datos, el procesado, almacenaje y posterior visualización de los resultados para que el cliente pueda determinar si se cumplen estos slas y pueda llevar a cabo acciones al respecto.

Este marco tecnológico se aplicará a un ejemplo real de una empresa de telecomunicaciones que tiene varios contratos slas con distintos clientes, utilizando tickets de operaciones de red que proporcionará la empresa NTT Data S.L.U, y que serán los que nutran todo el proceso. Este proceso de automatización de cálculo se realizará sobre los datos de uno de esos contratos, al que denominaremos C001 a partir de ahora, sin embargo el diseño e implementación estará orientado a que en un futuro, esto pueda aplicarse y adaptarse a otros contratos similares.

La automatización de procesos proporciona a las empresas una mayor productividad, fiabilidad y eficiencia entre otros muchos beneficios. Permite reducir tiempo en realizar tareas repetitivas y tediosas, minimiza errores y permite realizar una mayor trazabilidad sobre ellos y sobretodo mejora la calidad de gestión de la empresa.

Para poder conseguir esto, será necesario realizar los siguientes pasos:

- Se deberá realizar un análisis previo de los datos disponibles (datos de entrada, dimensiones, etc.) para poder plantear la solución que mejor se adapte a ellos.
- Se deberán establecer los SLAs a cumplir y las métricas de rendimiento que se quieren recoger para su posterior visualización.
- Se diseñará e implementará un proceso de Ingesta de datos, que permita recoger y volcar estos en una BBDD, para su posterior cálculo.
- Se diseñará e implementará un proceso ETL que permita extraer los datos (previamente ingestados) de la BBDD, transformarlos y realizar las operaciones de cálculo necesarias y finalmente cargar y/o exportar la información procesada (en una BBDD y/o ficheros).
- Se crearán informes de los resultados obtenidos para poder dar soporte a la toma de decisiones de la empresa, mediante herramientas de business Intelligence.

Objetivos Académicos

Desde el punto de vista académico, los objetivos de este proyecto han sido aplicar y ampliar conocimientos adquiridos durante el grado de Ingeniería Informática, como son:

- Conocimientos sobre Bases de Datos relacionales, adquiridos en asignaturas: *Tecnología y Diseño de Bases de Datos (TDBD)* y *Administración de Bases de Datos (ABD)*, gracias a las cuales he sido capaz de crear y administrar tablas y bases de datos en Maria DB.

- Mayor fluidez en el uso de comandos Linux para la creación de entornos, instalación de librerías, programación de tareas con cron, etc. Adquiridos en las asignaturas de sistemas operativos como *Estructura de Sistemas Operativos (ESO)* o *Administración de sistemas Operativos (ASO)*.

- Aplicación de técnicas y herramientas de inteligencia de negocio utilizadas actualmente por las empresas, gracias a la asignatura de *Sistemas de Información y Dirección de Organizaciones (SIDO)*, en la cual aprendí sobre Power BI, Azure, Sharepoint y otras tecnologías.

- Por último, la asignatura *Planificación y Gestión de Plataformas Informáticas (PGPI)* ha permitido realizar una planificación de las tareas y costes de este proyecto.

1.3. Estado del arte

1.3.1. ETLs

Los procesos ETL hacen referencia a un conjunto de técnicas, herramientas y tecnologías que forman parte de la integración de datos y que permiten obtener una utilidad de estos. La palabra ETL corresponde a las siglas en inglés Extraer, Transformar y Cargar, lo que significa que todo proceso ETL consta de estas tres fases. Extraer información de uno o varios orígenes de datos, transformar dicha información para adaptarla a las necesidades de negocio y finalmente cargarla en otro sistema para que pueda ser accesible por otros niveles de la organización que lo requieran.

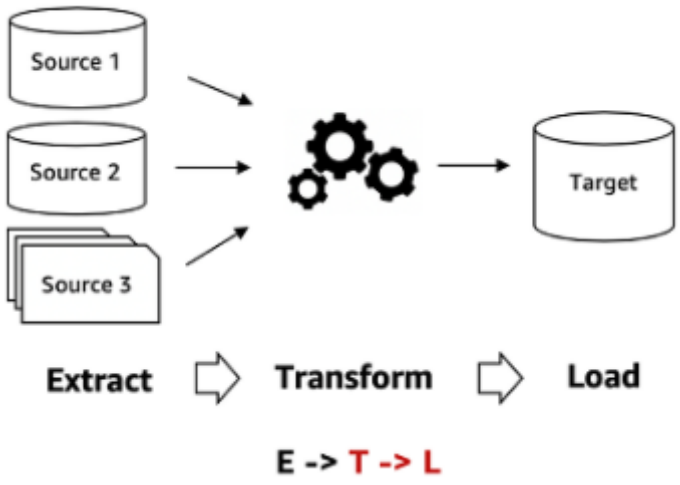


Figura 1.1: Proceso ETL

En una organización la importancia de una ETL es directamente proporcional a la dependencia que esta tiene del almacenamiento de datos. Todas las empresas almacenan grandes cantidades de datos de sus operaciones diarias, pero estos datos son recogidos en un estado bruto y no pueden ser leídas o analizadas directamente por las personas.

La información que recopilan las organizaciones es de gran interés para sus objetivos comerciales, pero esta no tiene significado por si sola. Para que los datos almacenados por la empresa tengan algún valor, deben ser procesados y transformados a un estado legible y práctico. Los procesos ETL juegan un papel importante en la integración de datos inteligentes para la posterior toma de decisiones del Business Intelligence (Inteligencia de Negocios). [27]

A continuación, se definen varios escenarios donde se refleja la importancia de utilizar los ETL en el ámbito de las organizaciones [27]:

- **ETL y el Almacenamiento de datos:** El almacenamiento de datos es un proceso complicado y costoso ya que requiere integrar, reorganizar y consolidad grandes volúmenes de datos que se encuentran en distintos sistemas, con el objetivo de mantener una fuente de datos única que pueda utilizarse en la inteligencia de negocios. Los ETL son procesos clave para recopilar, leer y migrar datos empresariales que se encuentran dispersos en múltiples fuentes en un único repositorio de datos, convirtiendo estos datos sin procesar en un conjunto de información homogeneizado. Además, si se encuentra configurado para ello, los ETL permiten realizar esto en tiempo real facilitando la toma de decisiones mediante en análisis de datos actualizados.

- **ETL y la Migración de datos:** [8] Los procesos ETL permiten trasladar datos de un lugar a otro, por lo que son herramientas clave para trasferir datos entre sistemas, por su eficacia y porque permiten minimizar los riesgos que suponen la migración de datos. Los datos son activos esenciales para una organización y es vital asegurarse que el proceso migratorio se complete con éxito. La migración de datos en una organización puede deberse a distintos factores: una actualización de software o implementación de uno nuevo, una actualización de bases de datos o migración hacia un nuevo data warehouse, etc.

- **ETL y la Calidad de los datos:** Como se ha visto, las organizaciones obtienen datos desde múltiples entradas por lo que es necesario procesar aquellos que realmente aporten valor. La calidad de datos es crucial, no sólo para garantizar la efectividad de un proceso en sí, sino para asegurar una coherencia en la toma de decisiones que se base en estos datos. Los procesos ETL proporcionan unas condiciones óptimas de calidad, realizando operaciones de limpieza, validación, perfilado y filtrado de datos, y almacenando estos de manera consistente.

FASES DE UN PROCESO ETL [23]

1. ***Fase de Extracción:*** Esta primera fase tiene como principal objetivo sintetizar toda la información necesaria en una o varias estructuras en las que se almacenen los datos normalizados, optimizándolos para la siguiente fase de transformación.

   En la mayoría de los casos, los datos que deben extraerse proceden de distintos orígenes o fuentes. Estas pueden ser internas (un CRM, ERP, ficheros almacenados en la nube, ficheros xml, etc.) o externas (web services de otros colaboradores o clientes, ficheros de bases de datos abiertos, etc) y también pueden ser datos que contengan valores nulos, datos erróneos, caracteres inválidos, registros

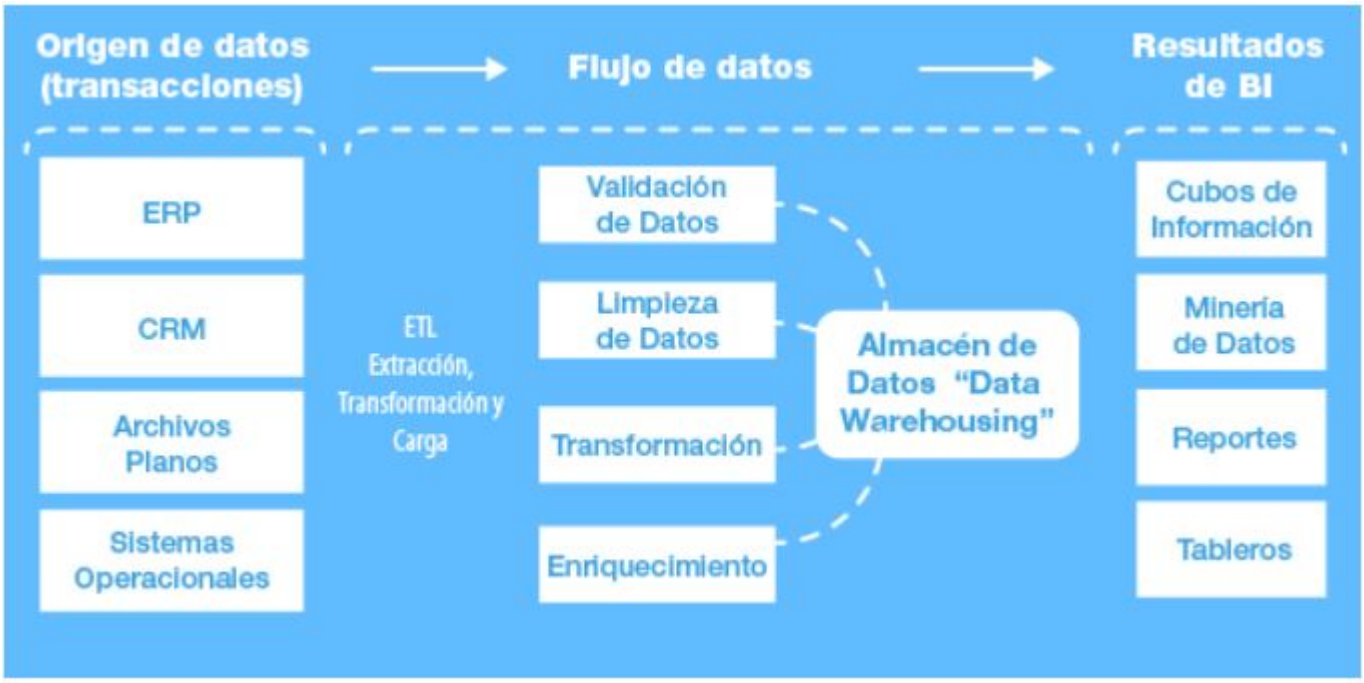


Figura 1.2: Flujo de un proceso ETL

duplicados y demás problemas. Cada origen de datos puede tener una estructura propia de almacenamiento y por ello esta fase debe encargarse de dejar los datos completamente homogeneizados, organizados y preparados para la siguiente fase, llegando a realizar alguna transformación.

2. ***Fase de Transformación:*** En esta fase se aporta valor a los datos y es donde se gestionan cuestiones técnicas. La fase de transformación aplica una serie de reglas de negocio o funciones sobre los datos extraídos, se encarga de descartar todos aquellos datos que no sean de interés, ya sea porque estén desfasados, sean incorrectos, irrelevantes o se encuentren duplicados. El objetivo es obtener datos útiles que estén listos para ser utilizados en la última fase. Entre las transformaciones más comunes podemos encontrar:

   - Eliminación de duplicados.
   - Estandarización de datos.
   - Limpieza de datos, solucionando anomalías, inconsistencias, etc.
   - Filtrado de columnas por distintas características
   - Obtención de nuevos valores calculados.
   - Generación de campos clave en el destino.
   - Agrupaciones de datos.
   - Combinación de datos de distintas fuentes.
   - Pivotaje de tablas.
   - División de columnas.

3. ***Fase de Carga:*** En esta tercera y última fase se almacenan los datos procesados de la fase anterior en el sistema destino. Las acciones que se realizaran en esta fase dependen de cada organización y sus necesidades. Lo más habitual es que la base de datos destino del proceso ETL sea una de las que ya se estén utilizando en la organización, por tanto mantendrá versionas antiguas de dicha base de datos. De esta manera la carga suele realizarse en paralelo, manteniendo ambas versiones, la original y la actualizada.

Cada una de estas fases puede desarrollarse de forma integrada, sobre todo cuando los procesos ETL no tienen demasiada complejidad. Sin embargo, existen ciertas ventajas a la hora de desarrollar cada fase de forma flexible e independiente [10]:

- Se puede asignar equipos distintos a cada fase y acotar plazos de trabajo.
- Cada fase puede desarrollarse utilizando distintas tecnologías
- Los cambios que afecten a una fase pueden aislarse del resto.
- Pueden realizarse pruebas en cada etapa por separado, afectando únicamente a la fase en la que se realicen.
- Los procedimientos desarrollados en las fases de extracción y carga pueden reutilizarse para otros procesos ETL.
- Se pueden reutilizar
- La capa de negocio puede implicarse en la fase de transformación para definir la lógica ETL, sin necesidad de involucrarse en cuestiones técnicas de la capa de extracción.

1.3.2. Acuerdos a Nivel de Servicio - SLA

Un SLA o Service Level Agreement es un contrato de servicios en el que se reflejan de forma detallada y concisa las condiciones de la relación entre un proveedor de servicios y sus clientes internos o externos. En dicho contrato se recogen desde los objetivos de este, características de los servicios contratados, responsabilidades de ambas partes, tanto del cliente como del proveedor, hasta finalmente las acciones en caso de su incumplimiento. Estos acuerdos son habituales a la hora de contratar servicios TI y en el outsourcing o externalización de servicios.

Originalmente, el término SLA se ha relacionado con compañías que suministran agua y electricidad, ya que estas utilizan este tipo de contratos con sus clientes. Sin embargo, el término "service level agreement"se ha popularizado en los últimos tiempos gracias a empresas de informática y telecomunicaciones [13], ya que estas tienen una cantidad de servicios muy flexibles y versátiles y es muy difícil realizar un control de calidad de estos.

El uso de acuerdos de nivel de servicio no se limita a los clientes externos de la empresa, también se usa de forma habitual a un nivel interno. Particularmente las organizaciones de TI utilizan los SLA con clientes internos, usuarios de otros departamentos dentro de la empresa. Estos acuerdos garantizan que se presten servicios dentro del negocio, bajo unas condiciones preestablecidas, permitiendo garantizar un mayor nivel de eficiencia y agilidad a la hora de abordar procesos y tareas [18]. Un departamento de TI crea un SLA para poder medir y justificar sus servicios y en ocasiones, compararlos con los de otros proveedores externos.

Tipos de SLA

Existen tres tipos de SLA utilizados en las organizaciones, y definidos por el ITIL [22]:

(A) *SLA de servicio:* Aplica un SLA estándar a todos los clientes que contratan un mismo servicio. El SLA depende de estándares que no cambian, lo que lo hace simple y directo para los proveedores. Es útil cuando nuestra empresa ofrece varios servicios con tiempos de resolución y respuesta diferentes.

(B) *SLA basado en el cliente:* Este tipo de acuerdo aplica a todos los servicios contratados por un mismo cliente, grupo de clientes o área de una empresa. Se trata de un contrato que recoge las necesidades, características y expectativas de un cliente en particular, por lo que no será posible replicarlo para otro cliente.

(C) *SLA Multinivel:* Combina los anteriores y aplica a nivel corporativo en todas las áreas de una organización. En dicho acuerdo, el usuario final tiene la posibilidad de personalizar el SLA de acuerdo a sus necesidades, incluyendo nuevas condiciones. Los SLA Multinivel evitan duplicaciones o errores entre varios acuerdos.

Contenido de los SLAs

Todo acuerdo SLA debe incluir tres partes fundamentales en las que se engloban todos los contenidos que definen de forma completa un acuerdo de servicio entre cliente y proveedor de servicios [18]:

Figura 1.3: Contenido de un acuerdo a nivel de Servicio

1. **Declaración de objetivos**. Los objetivos son una parte indispensable en cualquier contrato, por ello un acuerdo SLA debe incluir cuales son los objetivos del mismo.

2. **Especificación de los servicios** [24]. Deben incluirse una lista detallada de aquellos servicios que se incluyen y excluyen del acuerdo, para poder definir las condiciones de prestación de servicio. Además deben incluirse condiciones de disponibilidad del servicio y las obligaciones y responsabilidades de cada parte. Por un lado, las responsabilidades del proveedor a la hora de dar un servicio y por otro las del cliente a la hora de utilizarlo.

3. **Gestión del servicio** [24]. Relacionados con este apartado, debe incluirse información sobre los métodos y estándares de medición, así como el tipo de informes que se utilizarán, las condiciones y frecuencia de las mediciones, el proceso de resolución de disputas, medios para solucionar problemas de servicio, una cláusula de indemnización y las opciones para actualizar, renovar o finalizar dicho acuerdo.

Métricas

Existen distintas métricas que permiten medir el rendimiento y controlar si se está ofreciendo el servicio correcto o si se están cumpliendo o no las condiciones pactadas. Las métricas y condiciones incluidas en cada SLA difieren mucho dependiendo del tipo de servicio que se ofrece, sin embargo, la más habitual y que se incluye en la mayoría de los acuerdos, es la Disponibilidad del servicio. Esta métrica mide el tiempo que el servicio está funcionando y es accesible para el cliente, es una métrica muy habitual y su valor suele representarse con un porcentaje. Sin embargo, podemos identificar cuatro tipos principales de métricas a incluir en un SLA, que nos permiten estructurar correctamente el acuerdo, estas son [18]:

- **Métricas de Rendimiento:** aquellas en las que se compara periódicamente el rendimiento especificado con el rendimiento real obtenido.
- **Métricas de velocidad de respuesta:** son aquellas que miren la respuesta del proveedor cuando se realiza una petición, se completa una tarea o se resuelve un problema o incidencia. Entre estas, las métricas más habituales son:
  - Tiempo de Respuesta: tiempo que tarda el proveedor en responder a una solicitud o problema del cliente.
  - Tiempo de Resolución: tiempo que tarda en resolverse un problema una vez que lo registra el proveedor de servicio.
- **Métricas de Eficiencia:** miden la capacidad que tiene el proveedor para ofrecer un servicio efectivo a un coste razonable.
- **Métricas de calidad de trabajo:** miden si el servicio ofrecido cumple los requisitos y especificaciones establecidos previamente.

KPI (Key Performance Indicator)

Existe otra herramienta que ayuda a las organizaciones a conocer y medir el desempeño de individuos, unidades de negocio, proyectos, etc. en función de los objetivos estratégicos. Estos son los KPIs, indicadores clave de rendimiento, y pueden formar parte de un SLA para medir la entrega de los estándares de servicio definidos [9]. Ambos cumplen roles diferentes que mejorarán la calidad en servicios de la empresa.

Mientras que los SLAs definen el convenio general y los estándares de servicio entre proveedores y clientes, los KPIs se utilizan para monitorear los niveles de rendimiento. Los primeros son documentos más generales que describen las características de los acuerdos, mientras que los segundos miden el desempeño de las empresas en relación a sus objetivos estratégicos [7].

1.3.3. Sistema de Ticketing

El cumplimiento de los SLA es una actividad esencial para cualquier empresa de TI. En ocasiones, debido a la cantidad de proyectos, cargas de trabajo y el manejo de incidencias o emergencias, el seguimiento de los cumplimientos de estos acuerdos se puede ver afectado e incluso pasarse por alto [30].

Las empresas deben cumplir los plazos estipulados y satisfacer las expectativas de sus clientes. Una forma de gestionar adecuadamente los acuerdos SLA es mediante un sistema de ticketing, ya que permite mejorar los flujos de trabajo, brindar atención directa y transparente, mantener los estándares de calidad y mejorar la experiencia de los clientes [25]. Sobretodo, un sistema de ticketing permite identificar cuando es probable que un ticket incumpla un SLA.

Una herramienta de ticketing es un sistema que permite a las empresas gestionar y procesar las solicitudes y peticiones de sus clientes y les ayuda a resolver sus incidencias. Este tipo de sistemas son muy utilizados por los departamentos de SAT (Servicio Asistencia Técnica) [28], ayudándoles a optimizar el tiempo de resolución de incidencias y mejorar la experiencia del cliente. El objetivo de estos sistemas es que la comunicación entre cliente y empresa sea fluida, transparente y dinámica.

Funcionamiento de una herramienta o sistema de tickets

Un ticket de soporte es un registro en el cual se almacena toda la información relevante de un incidente o solicitud entrante de un cliente [25], como los datos del cliente, detalles del problema, fecha en la que ocurrió, etc. Al abrir un nuevo ticket, se le asigna una prioridad, dependiendo de la urgencia o el asunto y de acuerdo a las políticas de la empresa.

Las herramientas de ticketing permiten categorizar las consultas de los clientes, ayudando a las empresas a organizar los flujos de trabajo y priorizar aquellos que requieren actuación inmediata. Esto también ayuda a mantener una buena relación con el cliente, ya que, aunque su principal preocupación sea recibir respuestas instantáneas y soluciones rápidas, también necesita poder realizar un seguimiento de sus consultas y que le mantengan informado. Las herramientas de ticketing ayudan a identificar clientes para brindar atención personalizada.

Generalmente, la mayoría de estos sistemas, asignan ciertos estados a los tickets que permiten categorizar los incidentes. Estos pueden actualizarse tantas veces como sea necesario hasta que se resuelvan:

- Al recibir una nueva solicitud de un cliente, el sistema de tickets convierte esta en un ticket, que se cataloga automáticamente como '*Nuevo*'.
- El siguiente paso será asignar un agente a dicho ticket de soporte, que será el encargado de resolverlo. Una vez asignado, el estado del ticket pasará a ser '*Abierto*'.
- Durante su resolución, el agente puede necesitar nueva información del cliente. En esta caso el estado del ticket cambia a '*Pendiente*' hasta que el cliente proporcione dicha información y se retome su estado '*Abierto*'. En caso de no responder en un periodo de tiempo, es posible que los tickets se cierren automáticamente.
- De forma similar al anterior, existe un estado interno en el cual un agente puede necesitar más información para poder resolver el ticket, pero en esta ocasión es un miembro del equipo quien debe proporcionarla. Este estado es '*En Espera*' y el cliente no puede acceder a esta información.
- Por último, cuando el agente resuelve el ticket y el cliente recibe una solución, el estado de este será '*Concluido o Solucionado*'.

A partir del momento en que se recibe un nuevo incidente y se genera un nuevo ticket de soporte, este sistema asocia el cliente que realiza la solicitud con el contrato SLA que tenga con la empresa, para garantizar que los agentes cumplan con las condiciones pactadas [25].

Una de las grandes ventajas de utilizar herramientas de tickets es, como ya hemos comentado anteriormente, que permite mejorar el cumplimiento de los acuerdos de nivel de servicio (SLA). Estas permiten conocer cuando se esta inclumpliendo un SLA para algún ticket y proporciona estadísticas de cumplimiento que reflejan tendencias y pautas [29]. Sin embargo, el uso de estos sistemas también proporciona los siguientes beneficios:

- Optimiza el trabajo de los agentes y la comunicación entre ellos.
- Incrementa la productividad. Automatizar el desvío de solicitudes bajo unos criterios preestablecidos permite reducir tiempo en tareas manuales y acelera la toma de decisiones.
- Permite realizar una mayor trazabilidad de los casos abiertos, evitando que alguna solicitud pueda quedarse sin resolver.
- Detección rápida de fallos. Se crea un banco de datos que facilita la correlación entre problemas anteriores y sus posibles soluciones.
- Permite priorizar los casos más importantes. Los tickets ayudan a clasificar y ordenar los casos más urgentes.
- Reduce el tiempo de respuesta y resolución de problemas.
- Mejoran la experiencia del cliente y su satisfacción, lo que aumenta la posibilidad de fidelización de estos y mejora la imagen de la empresa.
- Ayuda la empresa a recopilar información e identificar problemas en los procedimientos que llevan a cabo y en general identificar fortalezas y debilidades para mejorar la toma de decisiones.

Capítulo 2

Planificación del Proyecto

2.1. Planificación de la Gestión de Riesgos

La gestión de riesgos es uno de los procesos que deben llevarse a cabo en las fases tempranas de un proyecto, más concretamente durante la fase de planificación. Esta práctica consiste en identificar, analizar y responder de manera proactiva a diferentes tipos de riesgos potenciales de un proyecto [11]. Conocer con antelación cuáles son estos riesgos potenciales ayuda a prevenirlos, a establecer unos objetivos del proyecto razonables y a elaborar una buena planificación [2].

Un riesgo en un proyecto es un evento o condición incierta que, en caso de suceder, puede afectar a alguno de los objetivos del proyecto. Los riesgos pueden ser sucesos que causen retrasos en el cronograma del proyecto, que causen excesos en el presupuesto previsto o cualquier causa que disminuya el rendimiento del equipo. La gestión de riesgos nos permite categorizar aquellos que tienen mayor probabilidad de afectar nuestro proyecto y elaborar un plan para mitigarlos en caso de que se materialicen.

El primer paso en la gestión de riesgos será identificarlos y analizar cuál será la probabilidad de que se materialicen y su impacto en el proyecto. Además, una vez identificados debemos asignarlos un nivel de prioridad y crear un plan de respuesta en caso de que ocurran. A la hora de priorizar los riesgos debemos tener en cuenta estos aspectos anteriores, la probabilidad y su impacto, para poder elaborar una planificación más efectiva y poder mitigar aquellos riesgos que puedan tener mayor impacto en nuestro proyecto.

2.1.1. Identificación de Riesgos

Para poder identificar los riesgos se han establecido 4 categorías diferentes en las que clasificar cada uno de ellos:

1. **Tecnológico:** aquellos relacionados con los equipos informáticos utilizados por el equipo para desarrollar el proyecto.
2. **Personal:** relacionados directamente con los miembros del equipo de trabajo. Dada la naturaleza del proyecto, existe un único trabajador/alumno que dedica tiempo al proyecto.
3. **Planificación:** relacionados con la planificación previa del proyecto.
4. **Implementación:** son aquellos riesgos que pueden aparecer durante la implementación de cada fase o al finalizar estas.

En la tabla 2.1 se muestra una lista con los riesgos identificados.

ID	Nombre del Riesgo	Categoría
R01	Fallo del sistema Informático	Tecnológico
R02	Problemas con el proveedor de servicios de Internet	Tecnológico
R03	Problemas en la instalación de alguna herramienta	Tecnológico
R04	Pérdida de información	Tecnológico
R05	Bajo Rendimiento	Personal
R06	Indisposición del personal	Personal
R07	Aprendizaje de nuevas herramientas o lenguajes de programación	Planificación
R08	Retraso de tareas en cascada	Planificación
R09	Estimación errónea de tiempos y esfuerzo	Planificación
R10	Estimación errónea de presupuestos	Planificación
R11	Falta de seguimiento y control	Planificación
R12	Producto implementado no cumple los objetivos	Implementación
R13	No se puede implementar alguna funcionalidad	Implementación
R14	Obsolescencia de software o funcionalidades de herramientas	Implementación
R15	El producto final tiene fallos	Implementación

Tabla 2.1: Identificación de Riesgos

2.1.2. Análisis y Priorización

Una vez identificados los riesgos potenciales, el siguiente paso será realizar un análisis y priorizar aquellos que tendrán mayor probabilidad de que sucedan y cuyo impacto será mayor. Además, elaboraremos un plan de respuesta para cada uno de ellos. Es importante realizar este análisis para no perder tiempo mitigando aquellos riesgos que no tendrán apenas impacto sobre nuestro proyecto.

Para poder evaluar los riesgos de manera clara y eficaz se va a crear una matriz de riesgos en base a dos ejes principales [1], Probabilidad (frecuencia con que ocurrirá un evento) e Impacto (relacionado con la severidad), como se viene mencionando. Para tener mayor precisión se han establecido 5 niveles distintos para cada uno de los ejes y se les ha asignado un valor numérico del 1 al 5, de manera que obtendremos una calificación de cada riesgo según el punto de intersección de ambos ejes [6].

Eje X - Probabilidad

- **Raro:** La probabilidad de que ocurra el evento es extremadamente baja. Valor numérico: 1
- **Bajo/improbable:** Existen pocas probabilidades de que ocurra el evento. Valor numérico: 2
- **Medio/Posible:** La probabilidad de que ocurra es ocasional. Es necesario planear acciones pero no urgente. Valor numérico: 3
- **Alto/Probable:** Existe una gran probabilidad de que ocurra y/o ocurre con frecuencia. Es necesario planear acciones para mitigar sus consecuencias. Valor numérico: 4
- **Casi segura:** Es un riesgo inevitable y es necesario pensar su impacto y las acciones a llevar a cabo en caso de que ocurra. Valor numérico: 5

Eje Y - Impacto

- **Insignificante:** No provoca ningún problema y puede ser ignorado, a no ser que ocurra con demasiada frecuencia. Valor numérico: 1
- **Leve:** Tiene muy poco impacto sobre el proyecto y/o sus objetivos, imperceptible. V. numérico: 2

- **Medio:** Causa ciertos problemas que pueden ser corregidos, pero pueden tener impacto a medio/largo plazo. Valor numérico: 3
- **Severo:** Compromete los resultados del proyecto, provocando retrasos considerables. V. numérico: 4
- **Muy Grave:** Puede paralizar el desarrollo del proyecto o tener consecuencias irreversibles. Valor numérico: 5

Muy Grave	5	10	15	20	25
Severo	4	8	12	16	20
Medio	3	6	9	12	15
Leve	2	4	6	8	10
Insignificante	1	2	3	4	5
	Rara	**Baja**	**Media**	**Alta**	**Certera**

Tabla 2.2: Matriz de riesgos

IDR01: Fallo del sistema Informático

Descripción: Problemas con alguno de los dispositivos electrónicos utilizados para realizar el proyecto, que ocasionan retrasos en la realización de alguna de las tareas y a su vez retrasos en el cronograma. Este evento provocaría un retraso de 1/2 días, dependiendo del tipo de problema o fallo.

- **Probabilidad:** Baja (2)
- **Impacto:** Severo (4)
- **Nivel de Riesgo:** 8

Solución: Disponer de algún equipo informático de repuesto.

IDR02: Problemas con el proveedor de servicios de Internet

Descripción: Problemas con el proveedor de servicios de Internet que impide que los sistemas informáticos se conecten a la red de Internet o lo hagan más lentamente de lo habitual. Este evento apenas tendrá impacto ya que de suceder

- **Probabilidad:** Rara (1)
- **Impacto:** Insignificante (1)
- **Nivel de Riesgo:** 1

Solución: A priori se ignorará, ya que durante el periodo en el que dure el problema se podrán buscar alternativas (uso de datos móviles, realización de tareas que no impliquen acceso a internet, búsqueda de lugares con wifi pública, etc) pero si el problema persiste se deberá hablar con el proveedor o negociar con uno nuevo.

IDR03: Problemas en la instalación de alguna herramienta

Descripción: la instalación de algún software nuevo se demora más de lo habitual debido a falta de espacio en el equipo, incompatibilidad de versiones, falta de permisos, etc. Esto provoca retrasos en la realización de alguna de las tareas principales.

- **Probabilidad:** Baja (2)
- **Impacto:** Medio (3)
- **Nivel de Riesgo:** 6

Solución: Se deberá adaptar la planificación lo antes posible para minimizar las consecuencias.

IDR04: Pérdida de información

Descripción: Pérdida de archivos, datos de la BBDD, de procesos, etc. Sea cual sea su motivo, robo, malware, accidente, descuidos, etc. Esto provoca retrasos en el cronograma.

- **Probabilidad:** Media (3)
- **Impacto:** Muy Grave (5)
- **Nivel de Riesgo:** 15

Solución: Realizar copias de seguridad periódicas, sobretodo de las BBDD, almacenar los archivos originales o versiones de estos en la nube y mantener los sistemas de seguridad actualizados. En caso de que esto suceda, se deberá adaptar la planificación lo antes posible para minimizar las consecuencias.

IDR05: Bajo Rendimiento

Descripción: El alumno, único miembro del equipo de trabajo, no rinde como debería, sea cual sea el motivo. Esta situación provoca retrasos en la realización de actividades, no más de 1 día de retraso.

- **Probabilidad:** Media (3)
- **Impacto:** Leve (2)
- **Nivel de Riesgo:** 6

IDR06: Indisposición del personal

Descripción: Enfermedad, imprevistos o alguna situación personal que provocan que el miembro del equipo (alumno) no pueda dedicarle el tiempo esperado al proyecto. Dependiendo de la situación puede darse un retraso en las tareas de 2/3 días.

- **Probabilidad:** Casi Seguro (5)
- **Impacto:** Medio (3) o Severo (4) si sucede con mucha frecuencia.
- **Nivel de Riesgo:** 15/20

Solución: Durante la planificación, establecer una fecha de finalización prevista y otra contemplando el peor caso teniendo en cuenta posibles retrasos en nuestro cronograma. En caso de que este riesgo se materialice, se deberá adaptar la planificación.

IDR07: Aprendizaje de nuevas herramientas o lenguajes de programación

Descripción: El trabajador necesita disponer de más tiempo para aprender sobre un nuevo software o un lenguaje de programación desconocido, lo que provoca retrasos en el cronograma.

- **Probabilidad:** Casi Seguro (5)
- **Impacto:** Severo (4)
- **Nivel de Riesgo:** 20

Solución: Conocer de antemano cuál será el software y las herramientas que se van a utilizar, conocer el nivel de conocimiento previo del trabajador y programar con antelación un periodo para aprender y acostumbrarse a trabajar con ellas. En caso de que este riesgo se materialice, podemos consultar tanto a tutores académicos como tutores de la empresa de prácticas, y se deberá adaptar la planificación lo antes posible para minimizar las consecuencias.

IDR08: Retraso de tareas en cascada

Descripción: El retraso de una tarea provoca retrasos en cascada en tareas dependientes.

- **Probabilidad:** Alta (4)
- **Impacto:** Severo (4)
- **Nivel de Riesgo:** 16

Solución: Durante la planificación, establecer una fecha de finalización prevista y otra contemplando el peor caso teniendo en cuenta posibles retrasos en nuestro cronograma. En caso de que este riesgo se materialice, se deberá adaptar la planificación lo antes posible para minimizar las consecuencias.

IDR9: Estimación errónea de tiempos y esfuerzo

Descripción: La planificación de tareas inicial y la estimación del esfuerzo no se ha realizado correctamente, lo que provoca un retraso en el cronograma.

- **Probabilidad:** Alta (4), debido a la inexperiencia del único miembro del equipo.
- **Impacto:** Severo (4)
- **Nivel de Riesgo:** 16

Solución: Durante la planificación, establecer una fecha de finalización prevista y otra contemplando el peor caso teniendo en cuenta posibles retrasos en nuestro cronograma. En caso de que este riesgo se materialice, se deberá adaptar la planificación según sea necesario.

IDR10: Estimación errónea de presupuestos

Descripción: La planificación de presupuesto inicial no se ha realizado correctamente y han surgido imprevistos que aumentan el coste del presupuesto. Debido a la naturaleza académica del proyecto este riesgo no se contempla ya que no existe un presupuesto real.

- **Probabilidad:** Rara (1)
- **Impacto:** Insignificante (1)
- **Nivel de Riesgo:** 1

IDR11: Falta de seguimiento y control

Descripción: No realizar un seguimiento sobre las tareas y su duración puede causar que se desconozca el estado real del proyecto hasta que no se ha avanzado suficiente. Podemos tener una imagen errónea del estado en el que nos encontramos y como consecuencia desviarnos de la fecha de finalización establecida.

- **Probabilidad:** Media (3)
- **Impacto:** Severo (4)
- **Nivel de Riesgo:** 12

Solución: Planificar revisiones periódicas del proyecto al final de cada fase para actualizar las tareas y reconocer retrasos.

IDR12: Producto implementado no cumple los objetivos

Descripción: Un mal diseño o un diseño demasiado sencillo no cubre los objetivos y funcionalidades esperadas y se debe volver a diseñar e implementar.

- **Probabilidad:** Media(3)
- **Impacto:** Muy Grave (5)
- **Nivel de Riesgo:** 15

Solución: Poner especial hincapié en la definición de objetivos al inicio del proyecto y planificar revisiones para garantizar que no nos desviamos de su cumplimiento. En caso de que este riesgo se materialice, se deberá adaptar la planificación lo antes posible para minimizar las consecuencias.

IDR13: Imposibilidad para implementar alguna funcionalidad

Descripción: Una o varias de las funcionalidades diseñadas no se puede llegar a implementar con las herramientas y lenguajes utilizadas o con los conocimientos actuales del miembro del equipo y se deben buscar alternativas para resolverlo o invertir tiempo en conocer más a fondo dichas herramientas o lenguajes, lo que provocará retrasos en el cronograma.

- **Probabilidad:** Alta (4). La probabilidad es alta ya que se están utilizando lenguajes y herramientas hasta el momento desconocidas por el alumno.
- **Impacto:** Severo (4)
- **Nivel de Riesgo:** 16

Solución: Dedicar un tiempo en la planificación para una curva de aprendizaje, donde conocer las nuevas herramientas y sus posibilidades. En caso de que este riesgo se materialice, podemos consultar tanto a tutores académicos como tutores de la empresa de prácticas, y se deberá adaptar la planificación lo antes posible para minimizar las consecuencias.

IDR14: Obsolescencia de software o funcionalidades de herramientas

Descripción: Tras haber implementado alguna funcionalidad del proyecto, alguna herramienta software ha quedado obsoleta o han variado sus políticas de uso y se deben buscar alternativas al uso de estas herramientas o funcionalidades y/o adaptar la funcionalidad afectada a las nuevas políticas. Esto produce retrasos en el cronograma.

- **Probabilidad:** Media(3)
- **Impacto:** Alto (4)
- **Nivel de Riesgo:** 12

Solución: Se deberá adaptar la planificación lo antes posible para minimizar las consecuencias.

IDR15: El producto final tiene fallos

Descripción: El producto final contiene fallos y se debe emplear tiempo en corregirlos y realizar nuevas pruebas, retrasando el cronograma.

- **Probabilidad:** Alta (4)
- **Impacto:** Severo (4)
- **Nivel de Riesgo:** 16

Solución: Planificar una batería de pruebas detallada al final de cada fase. En caso de que este riesgo se materialice, se deberá adaptar la planificación lo antes posible para minimizar las consecuencias.

2.1.3. Revisión

Una vez identificado, asignado un valor numérico y una acción correctiva a los riesgos encontrados, es necesario realizar un monitoreo durante el tiempo en que dure el proyecto para evaluar su estado y la efectividad de las acciones propuestas. Esto puede hacerse al finalizar cada una de las fases. Y se tendrá en cuenta la tabla 2.3, donde se han ordenado según el nivel de prioridad establecido en el apartado anterior.

Posición	ID	Nombre del Riesgo	Nivel de Riesgo
1	R07	Aprendizaje de nuevas herramientas o lenguajes de programación	20
2	R06	Indisposición del personal	15/20
3	R08	Retraso de tareas en cascada	16
	R09	Estimación errónea de tiempos y esfuerzo	16
	R15	El producto final tiene fallos	16
	R13	No se puede implementar alguna funcionalidad	16
4	R12	Producto implementado no cumple los objetivos	15
	R04	Pérdida de información	15
5	R14	Obsolescencia de software o funcionalidades de herramientas	12
	R11	Falta de seguimiento y control	12
6	R01	Fallo del sistema Informático	8
7	R03	Problemas en la instalación de alguna herramienta	6
	R05	Bajo Rendimiento	6
8	R02	Problemas con el proveedor de servicios de Internet	1
	R10	Estimación errónea de presupuestos	1

Tabla 2.3: Niveles de Riesgos

2.2. Planificación y estimación temporal.

Realizar una estimación global de las horas necesarias para desarrollar un proyecto y planificar los hitos temporales necesarios para ello es una tarea complicada. Por ello, se van a utilizar el número de créditos correspondientes al Trabajo de Fin de Grado para realizar una estimación de la cantidad de horas requeridas para ello. Puesto que un crédito ECTS equivale a 25 horas de trabajo [26] por parte del estudiante y el Trabajo Fin de Grado tiene asignados 12 créditos, se han estimado un mínimo de 300 Horas de dedicación al proyecto. Esta estimación de esfuerzo representa la cantidad de horas útiles necesarias para desarrollar el proyecto, sin embargo, se deben tener en cuenta posibles retrasos que pueda sufrir.

En primer lugar, se va a establecer un calendario de trabajo de acuerdo con la disponibilidad del alumno. Este calendario estará dividido a priori en dos tramos, el primero de ellos finalizará el día 31 de marzo de 2022 y el segundo durará desde entonces hasta la fecha de finalización del proyecto. Consideraremos este último como el calendario Predeterminado en MS Project, ya que será el de mayor duración. La única diferencia entre ambos periodos será que, en el primer tramo, los sábados y domingos se establecerán como 'No Laborables'. Estos dos tramos pueden verse en las tablas 2.4 y 2.5.

Día de la Semana	Horario de Mañana			Horario de Tarde			Total Horas
	Inicio	Fin	Horas	Inicio	Fin	Horas	
Lunes	No Laborable		0	17:00	21:00	4	4
Martes	No Laborable		0	18:30	20:30	2	2
Miércoles	No Laborable		0	18:30	20:30	2	2
Jueves	No Laborable		0	17:00	21:00	4	4
Viernes	No Laborable		0	17:00	21:00	4	4
Sábado	No Laborable		0	No Laborable		0	0
Domingo	No Laborable		0	No Laborable		0	0
						Total horas semanales	16

Tabla 2.4: Horario dedicado a Proyecto Hasta (31/03/2021) – Tramo 1

Día de la Semana	Horario de Mañana			Horario de Tarde			Total Horas
	Inicio	Fin	Horas	Inicio	Fin	Horas	
Lunes	No Laborable		0	17:00	21:00	4	4
Martes	No Laborable		0	18:30	20:30	2	2
Miércoles	No Laborable		0	18:30	20:30	2	2
Jueves	No Laborable		0	17:00	21:00	4	4
Viernes	No Laborable		0	17:00	21:00	4	4
Sábado	09:00	13:00	4	17:00	20:00	3	7
Domingo	09:00	13:00	4	No Laborable		0	4
						Total horas semanales	27

Tabla 2.5: Horario dedicado a Proyecto desde (01/04/2021) – Tramo 2 (Predeterminado)

Adicionalmente y atendiendo al calendario académico 2021-2022, se ha decidido añadir otro tramo para englobar aquellas fechas en las cuales el alumno, al no tener que asistir a clase en la Universidad, dispondrá de mayor tiempo para emplear en el proyecto. El horario de este nuevo tramo se puede visualizar en la tabla 2.6 y afectará únicamente a los lunes y miércoles por la mañana en los siguientes periodos:

- Semana Santa. Del 07 de abril de 2022 hasta el 17 de abril de 2022.
- Fin de la actividad lectiva del segundo cuatrimestre. Desde el 30 de mayo de 2022

Día de la Semana	Horario de Mañana			Horario de Tarde			Total Horas
	Inicio	Fin	Horas	Inicio	Fin	Horas	
Lunes	09:00	13:00	4	17:00	21:00	4	8
Martes	No Laborable		0	18:30	20:30	2	2
Miércoles	09:00	13:00	4	18:30	20:30	2	6
Jueves	No Laborable		0	17:00	21:00	4	4
Viernes	No Laborable		0	17:00	21:00	4	4
Sábado	09:00	13:00	4	17:00	20:00	3	7
Domingo	09:00	13:00	4	No Laborable		0	0
					Total horas semanales		27

Tabla 2.6: Horario dedicado a Proyecto – Tramo 3

De acuerdo con este calendario y habiendo establecido la fecha de inicio de proyecto el lunes 14 de marzo de 2022, se cumplirían 300 horas de proyecto el día 4 de junio de 2022. Sin embargo, se ha añadido un margen de tiempo para errores y posibles retrasos que pueda sufrir el proyecto. Este margen se ha establecido en 50 horas y situaría la fecha límite de finalización el 14 de junio de 2022. En la imagen 2.1 se puede ver la planificación óptima del proyecto, asignando 300 horas y finalizando en la fecha propuesta (Sin retrasos).

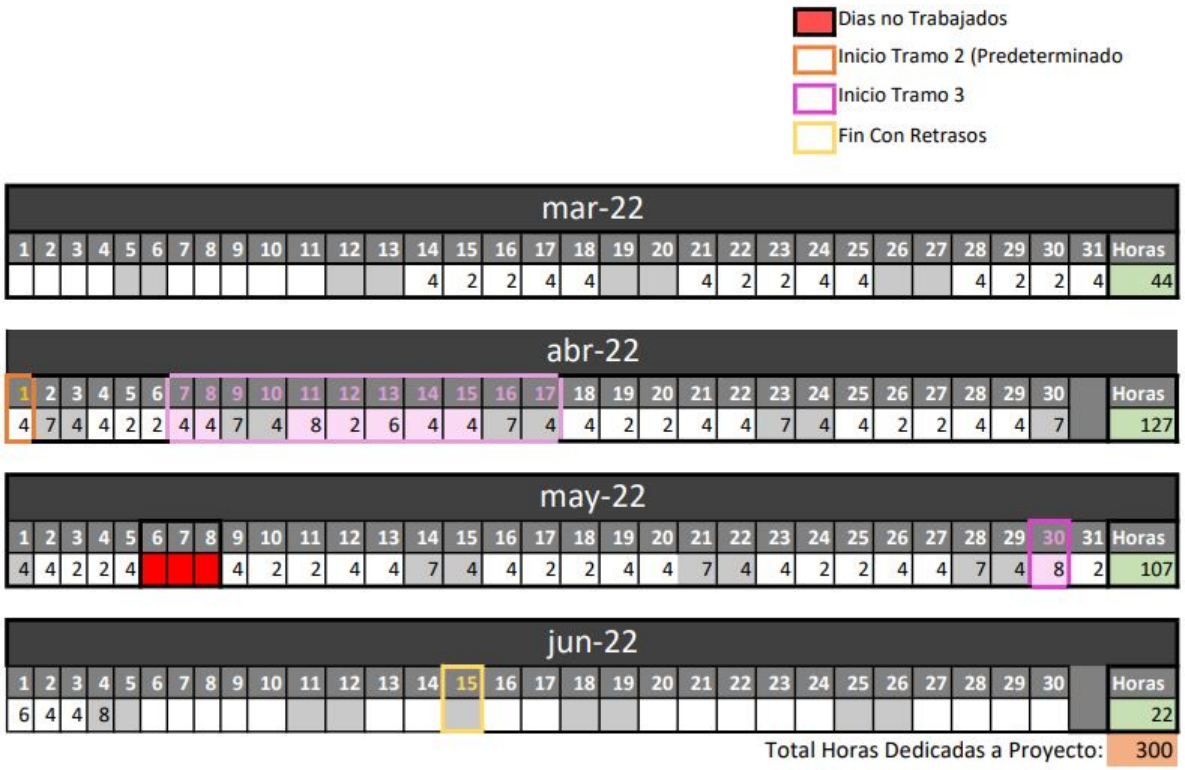


Figura 2.1: Horas dedicadas a proyecto por día

2.2.1. Planificación Inicial

Como se verá en el apartado 3 del siguiente capítulo, el software utilizado para la gestión del proyecto va a ser Microsoft Project. Los primeros pasos al crear un nuevo proyecto en esta herramienta serán, en primer lugar, introducir las fechas de comienzo y finalización del proyecto y en segundo, volcar el calendario de trabajo que hemos definido anteriormente. Microsoft Project permite crear calendarios específicos de proyectos, recursos y tareas individuales, reflejando los días y horas laborables. Además, se podrán marcar días específicos en los que no se vaya a trabajar por distintas razones, día festivo, enfermedad, fecha importante, etc. En este caso, se va a modificar el calendario estándar ya que únicamente contamos con un recurso que será el encargado de realizar todas las tareas definidas. La imagen 2.2 muestra la configuración de Microsoft Project para el Calendario Laboral.

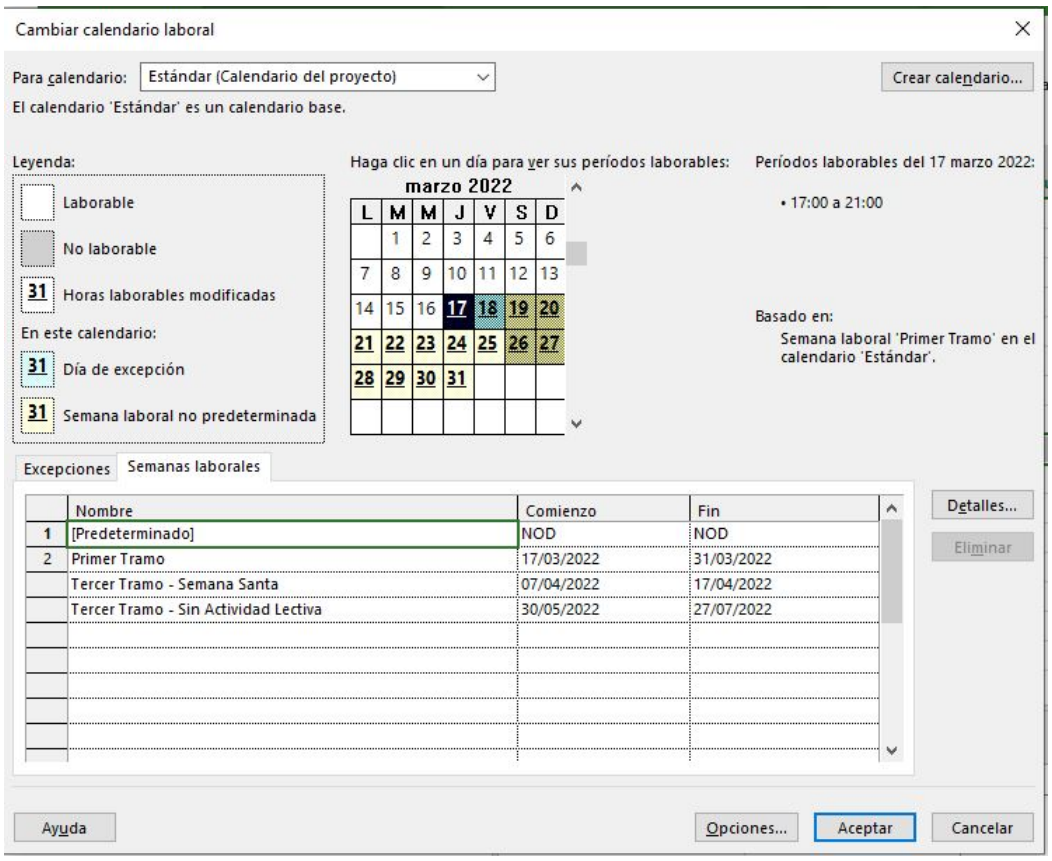


Figura 2.2: Calendario Laboral Microsoft Project

Una vez establecido nuestro horario laboral, se ha dividido el trabajo en 5 fases distintas que se detallarán continuación, y cuyo desglose de tareas se detalla en figura 2.3. Esta figura muestra la planificación inicial de tareas, llegando a cumplir las 300 horas estimadas y sin incluir retrasos sufridos.

1. **Fase Inicial:** En esta primera fase se han realizado tareas de preparación del trabajo futuro. En ella incluimos el estudio previo sobre el problema, interés que pueda tener, reuniones con miembros del equipo NTT Data para conocer la actividad que realizan y establecer un punto de partida, etc. Además, se incluye la elaboración y presentación de la propuesta de TFG, la definición de objetivos y la elaboración inicial del índice de contenidos y por último el estudio sobre el estado del arte.

2. **Fase de Planificación:** En esta fase se incluye todas aquellas tareas que han permitido estructurar el proyecto, desde el análisis y gestión de riesgos, la preparación del calendario de trabajo y estimación temporal, la planificación del presupuesto y la elicitación de requisitos. También incluimos el análisis

de los datos proporcionados por la empresa (para el posterior modelado y la creación de las métricas), el análisis sobre las herramientas a utilizar y tareas de aprendizaje o repaso sobre estas.

3. **Fase de Diseño:** se incluirán todas aquellas tareas que han permitido definir la solución propuesta y su arquitectura, desde la elaboración de los casos de uso (en este caso uno sólo), el modelo dimensional, el diseño de la BBDD y la estructuración de los datos y procesos implementados, hasta finalmente el diseño de los procesos de Ingesta, ETL y la visualización de datos.

4. **Fase de Implementación:** abarca tareas de preparación del entorno de trabajo, como la creación de entornos virtuales, tablas adicionales en la BBDD y el desarrollo del sistema a desarrollar, desde la ingesta hasta la visualización.

5. **Fase Final:** por último, se describen las tareas realizadas tras desarrollar el grueso del proyecto. Estas son, documentación de las pruebas realizadas, planteamiento de conclusiones y posibilidades futuras y un repaso final del proyecto.

Id	Nombre de tarea	Duración	Comienzo	Fin	Predecesoras
1	Comienzo Proyecto	0 horas	lun 14/03/22	lun 14/03/22	
2	**Fase Inicial**	**6,25 días**	**lun 14/03/22**	**sáb 02/04/22**	
3	Estudio Inicial y planteamiento del problema	20 horas	lun 14/03/22	lun 21/03/22	
4	Elaboración de propuesta	8 horas	vie 18/03/22	lun 21/03/22	3FF
5	Definición de Objetivos	10 horas	jue 17/03/22	lun 21/03/22	3FF
6	Elaboración del índice	2 horas	lun 21/03/22	lun 21/03/22	3FF
7	Estado del arte	30 horas	mar 22/03/22	sáb 02/04/22	3
8	**Fase de Análisis y Planificación**	**10,88 días**	**sáb 02/04/22**	**vie 22/04/22**	**2**
9	Análisis de Requisitos	8 horas	sáb 02/04/22	dom 03/04/22	
10	Preparación de Calendario y Estimación temporal	8 horas	dom 03/04/22	mar 05/04/22	9
11	Planificación de presupuesto	4 horas	lun 04/04/22	mar 05/04/22	10FF
12	Planificación y Gestión de Riesgos	12 horas	mié 06/04/22	sáb 09/04/22	11
13	Estudio sobre las Herramientas a utilizar	26 horas	sáb 09/04/22	mié 13/04/22	12
14	Análisis de los datos disponibles	8 horas	jue 14/04/22	vie 15/04/22	13
15	Aprendizaje NIFI - Realizacion de ejemplos	15 horas	sáb 16/04/22	vie 22/04/22	14
16	Repaso Power BI - Realizacion de ejemplos	10 horas	sáb 16/04/22	dom 17/04/22	14
17	**Fase de Diseño**	**7,75 días**	**vie 22/04/22**	**mié 11/05/22**	**8**
18	Elaboración de Caso de Uso	8 horas	vie 22/04/22	sáb 23/04/22	
19	Elaboración del Modelo Dimensional	12 horas	sáb 23/04/22	mar 26/04/22	18
20	Diseño de la BBDD	4 horas	mar 26/04/22	jue 28/04/22	19
21	Estructuración de Datos y Directorios	4 horas	mar 26/04/22	jue 28/04/22	20FF
22	Diseño Ingesta	12 horas	jue 28/04/22	sáb 30/04/22	20
23	Diseño ETL - Proceso de cálculo	14 horas	sáb 30/04/22	mié 04/05/22	22
24	Diseño Visualización	12 horas	mié 04/05/22	mié 11/05/22	23
25	**Fase Implementacion**	**10,38 días**	**mié 11/05/22**	**lun 30/05/22**	**17**
26	Creacion de entornos virtuales	2 horas	mié 11/05/22	jue 12/05/22	
27	Creación de tablas de la BBDD	4 horas	mié 11/05/22	jue 12/05/22	20
28	Implementación de la Ingesta	23 horas	mié 11/05/22	lun 16/05/22	22
29	Implementación de ETL - Proceso de cálculo	40 horas	lun 16/05/22	vie 27/05/22	28
30	Implementación de la Visualización	20 horas	vie 27/05/22	lun 30/05/22	29
31	**Fase Final**	**12,63 días**	**mié 11/05/22**	**vie 03/06/22**	
32	Inicio de la fase de pruebas	0 horas	mié 11/05/22	mié 11/05/22	25CC
33	Documentacion de pruebas	10 horas	lun 30/05/22	mié 01/06/22	25
34	Fin de la fase de pruebas	0 horas	mié 01/06/22	mié 01/06/22	33
35	Conclusiones y Trabajo futuro	6 horas	jue 02/06/22	vie 03/06/22	34
36	Repaso final	2 horas	vie 03/06/22	vie 03/06/22	35

Figura 2.3: Planificación inicial de tareas

2.2.2. Planificación Final y Riesgos Materializados

Tras la finalización de cada una de las fases se han realizado revisiones para identificar retrasos en las tareas producidos por la materialización de alguno de los riesgos detallados en el apartado 2.1.1 y se han reajustado estas tareas a los nuevos tiempos. La duración final del proyecto y de sus tareas se incluye en la siguiente imagen:

Id	Nombre de tarea	Duración	Comienzo previsto	Comienzo	Fin	Predecesoras
1	Comienzo Proyecto	0 horas	lun 14/03/22	lun 14/03/22	lun 14/03/22	
2	**Fase Inicial**	**7 días**	**lun 14/03/22**	**lun 14/03/22**	**dom 03/04/22**	
3	Estudio Inicial y planteamiento del problema	23 horas	lun 14/03/22	lun 14/03/22	mié 23/03/22	
4	Elaboración de propuesta	8 horas	lun 14/03/22	vie 18/03/22	mié 23/03/22	3FF
5	Definición de Objetivos	10 horas	jue 17/03/22	vie 18/03/22	mié 23/03/22	3FF
6	Elaboración del índice	2 horas	lun 21/03/22	mar 22/03/22	mié 23/03/22	3FF
7	Estado del arte	32 horas	mar 22/03/22	jue 24/03/22	dom 03/04/22	3
8	**Fase de Análisis y Planificación**	**11,25 días**	**dom 03/04/22**	**dom 03/04/22**	**sáb 23/04/22**	**2**
9	Análisis de Requisitos	8 horas	dom 03/04/22	dom 03/04/22	lun 04/04/22	
10	Preparación de Calendario y Estimación temporal	8 horas	mar 05/04/22	mar 05/04/22	jue 07/04/22	9
11	Planificación de presupuesto	4 horas	jue 07/04/22	jue 07/04/22	jue 07/04/22	10FF
12	Planificación y Gestión de Riesgos	12 horas	vie 08/04/22	vie 08/04/22	dom 10/04/22	11
13	Estudio sobre las Herramientas a utilizar	26 horas	dom 10/04/22	dom 10/04/22	vie 15/04/22	12
14	Análisis de los datos disponibles	8 horas	vie 15/04/22	vie 15/04/22	sáb 16/04/22	13
15	Aprendizaje NIFI - Realizacion de ejemplos	18 horas	sáb 16/04/22	sáb 16/04/22	sáb 23/04/22	14
16	Repaso Power BI - Realizacion de ejemplos	12 horas	sáb 16/04/22	sáb 16/04/22	mar 19/04/22	14
17	**Fase de Diseño**	**10 días**	**vie 22/04/22**	**mar 26/04/22**	**vie 20/05/22**	**8**
18	Elaboración de Caso de Uso	8 horas	sáb 23/04/22	mar 26/04/22	jue 28/04/22	
19	Elaboración del Modelo Dimensional	12 horas	dom 24/04/22	lun 02/05/22	jue 05/05/22	18
20	Diseño de la BBDD	6 horas	mié 27/04/22	lun 09/05/22	mar 10/05/22	19
21	Estructuración de Datos y Directorios	4 horas	mié 27/04/22	lun 09/05/22	mar 10/05/22	20FF
22	Diseño Ingesta	12 horas	vie 29/04/22	mié 11/05/22	sáb 14/05/22	20
23	Diseño ETL - Proceso de cálculo	14 horas	dom 01/05/22	sáb 14/05/22	lun 16/05/22	22
24	Diseño Visualización	12 horas	jue 05/05/22	mar 17/05/22	vie 20/05/22	23
25	**Fase Implementacion**	**13,5 días**	**mié 11/05/22**	**sáb 21/05/22**	**dom 12/06/22**	**17**
26	Creacion de entornos virtuales	4 horas	sáb 21/05/22	sáb 21/05/22	sáb 21/05/22	
27	Creación de tablas de la BBDD	4 horas	sáb 21/05/22	dom 22/05/22	dom 22/05/22	
28	Implementación de la Ingesta	28 horas	sáb 21/05/22	sáb 21/05/22	vie 27/05/22	22
29	Implementación de ETL - Proceso de cálculo	55 horas	sáb 28/05/22	sáb 28/05/22	lun 06/06/22	28
30	Implementación de la Visualización	25 horas	lun 06/06/22	lun 06/06/22	dom 12/06/22	29
31	**Fase Final**	**20,75 días**	**mié 11/05/22**	**sáb 21/05/22**	**jue 23/06/22**	
32	Inicio de la fase de pruebas	0 horas	sáb 21/05/22	sáb 21/05/22	sáb 21/05/22	25CC
33	Documentacion de pruebas	10 horas	dom 12/06/22	lun 20/06/22	mar 21/06/22	25
34	Fin de la fase de pruebas	0 horas	mar 14/06/22	mar 21/06/22	mar 21/06/22	33
35	Conclusiones y Trabajo futuro	6 horas	mié 15/06/22	mié 22/06/22	mié 22/06/22	34
36	Repaso final	2 horas	jue 16/06/22	jue 23/06/22	jue 23/06/22	35

Figura 2.4: Planificación Final de tareas

La planificación final muestra un desvío de 20 días respecto a fecha de finalización esperada del proyecto y de 8 respecto a la fecha estimada con retrasos. A continuación detallan los riesgos materializados y como afectaron a la planificación del proyecto:

En la *Fase Inicial* se materializó el riesgo *IDR08* en las Tareas 3 y 7, afectando a la fecha de finalización prevista y como consecuencia a aquellas tareas dependientes. En este caso afecto a la fecha de comienzo de la tareas 7 y 8 respectivamente.

Durante la *Fase de Análisis y Planificación*, se necesitó emplear más tiempo en las tareas de aprendizaje (15 y 16), produciéndose el riesgo IDR07.

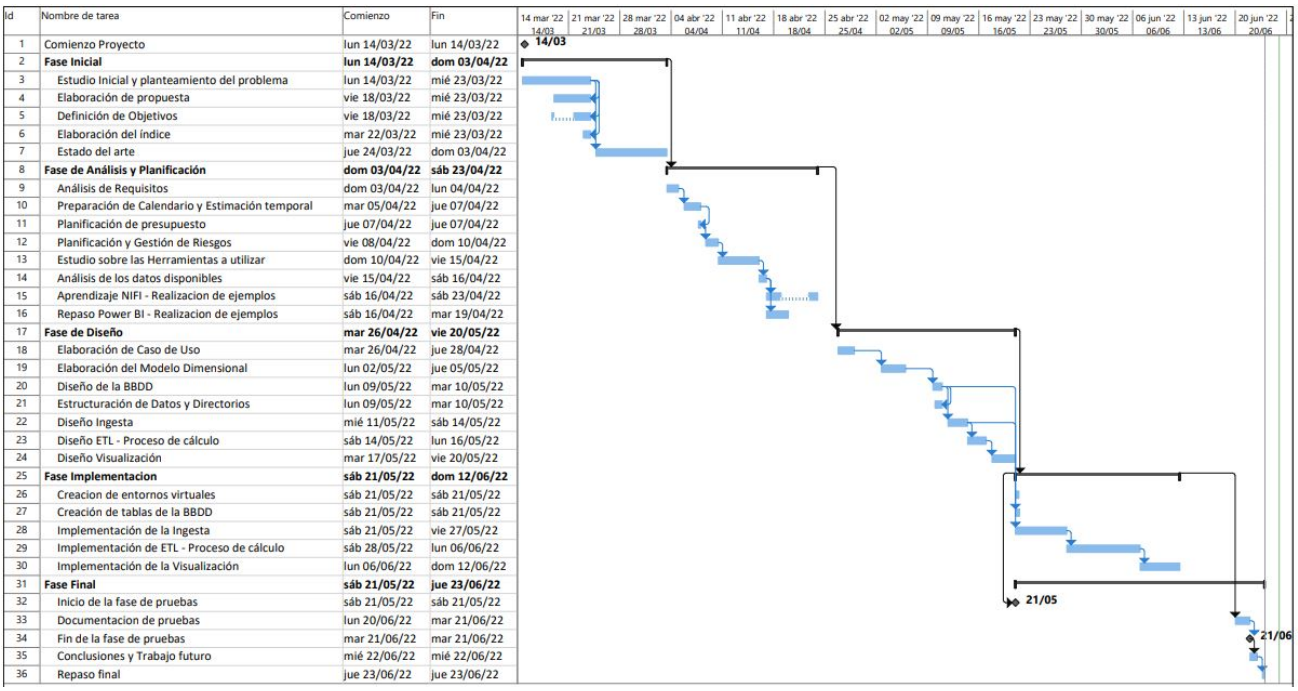
La *Fase de Diseño* tuvo varios retrasos relacionados con la disponibilidad, IDR05 y IDR06, debido a enfermedad e imprevistos personales.

Por último, en la *Fase de Implementación* el riesgo más presente fue el IDR09 en las tareas de implementación (28,29,30) puesto que se demoraron más tiempo de lo previsto. La instalación de entornos virtuales en Anaconda (IDR03) dio problemas y produjo un retraso en la finalización de la tarea. Además, se produjo una pérdida de información debido a un problema informático. Se perdieron los avances (IDR04) en la implementación de uno de los procesos antes de poder realizar una copia de seguridad.

Estos retrasos en la fase de implementación provocaron un retraso en cascada (IDR08) de la tarea 33 (Documentación de pruebas).

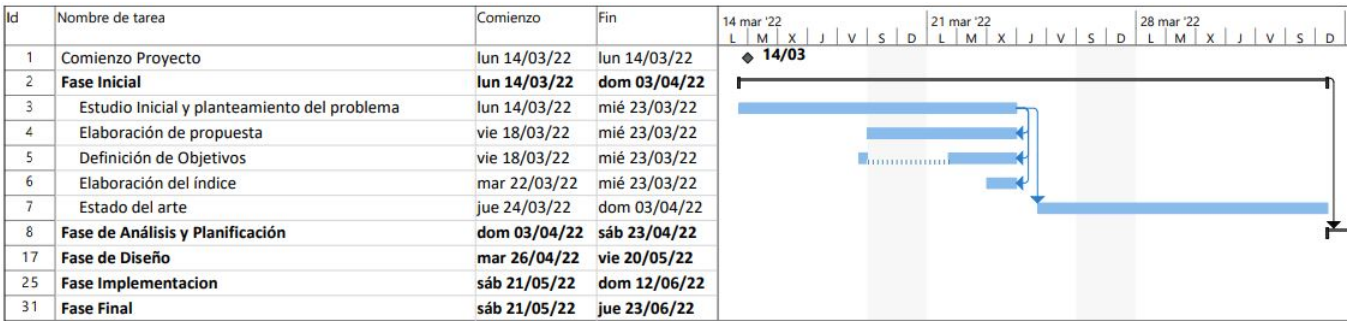
2.2.3. Diagrama de Gantt

A continuación se muestran los Diagramas de Gannt para cada una de las fases en las que se ha dividido el proyecto. Un diagrama de Gantt es una herramienta que proporciona una vista general de la duración de las tareas programadas.



Id	Nombre de tarea	Comienzo	Fin
1	Comienzo Proyecto	lun 14/03/22	lun 14/03/22
2	**Fase Inicial**	**lun 14/03/22**	**dom 03/04/22**
3	Estudio Inicial y planteamiento del problema	lun 14/03/22	mié 23/03/22
4	Elaboración de propuesta	vie 18/03/22	mié 23/03/22
5	Definición de Objetivos	vie 18/03/22	mié 23/03/22
6	Elaboración del índice	mar 22/03/22	mié 23/03/22
7	Estado del arte	jue 24/03/22	dom 03/04/22
8	**Fase de Análisis y Planificación**	**dom 03/04/22**	**sáb 23/04/22**
9	Análisis de Requisitos	dom 03/04/22	lun 04/04/22
10	Preparación de Calendario y Estimación temporal	mar 05/04/22	jue 07/04/22
11	Planificación de presupuesto	jue 07/04/22	jue 07/04/22
12	Planificación y Gestión de Riesgos	vie 08/04/22	dom 10/04/22
13	Estudio sobre las Herramientas a utilizar	dom 10/04/22	vie 15/04/22
14	Análisis de los datos disponibles	vie 15/04/22	sáb 16/04/22
15	Aprendizaje NIFI - Realizacion de ejemplos	sáb 16/04/22	sáb 23/04/22
16	Repaso Power BI - Realizacion de ejemplos	sáb 16/04/22	mar 19/04/22
17	**Fase de Diseño**	**mar 26/04/22**	**vie 20/05/22**
18	Elaboración de Caso de Uso	mar 26/04/22	jue 28/04/22
19	Elaboración del Modelo Dimensional	lun 02/05/22	jue 05/05/22
20	Diseño de la BBDD	lun 09/05/22	mar 10/05/22
21	Estructuración de Datos y Directorios	lun 09/05/22	mar 10/05/22
22	Diseño Ingesta	mié 11/05/22	sáb 14/05/22
23	Diseño ETL - Proceso de cálculo	sáb 14/05/22	lun 16/05/22
24	Diseño Visualización	mar 17/05/22	vie 20/05/22
25	**Fase Implementacion**	**sáb 21/05/22**	**dom 12/06/22**
26	Creacion de entornos virtuales	sáb 21/05/22	sáb 21/05/22
27	Creación de tablas de la BBDD	sáb 21/05/22	sáb 21/05/22
28	Implementación de la Ingesta	sáb 21/05/22	vie 27/05/22
29	Implementación de ETL - Proceso de cálculo	sáb 28/05/22	lun 06/06/22
30	Implementación de la Visualización	lun 06/06/22	dom 12/06/22
31	**Fase Final**	**sáb 21/05/22**	**jue 23/06/22**
32	Inicio de la fase de pruebas	sáb 21/05/22	sáb 21/05/22
33	Documentacion de pruebas	lun 20/06/22	mar 21/06/22
34	Fin de la fase de pruebas	mar 21/06/22	mar 21/06/22
35	Conclusiones y Trabajo futuro	mié 22/06/22	mié 22/06/22
36	Repaso final	jue 23/06/22	jue 23/06/22

Figura 2.5: Vista General de la Planificación



Id	Nombre de tarea	Comienzo	Fin
1	Comienzo Proyecto	lun 14/03/22	lun 14/03/22
2	**Fase Inicial**	**lun 14/03/22**	**dom 03/04/22**
3	Estudio Inicial y planteamiento del problema	lun 14/03/22	mié 23/03/22
4	Elaboración de propuesta	vie 18/03/22	mié 23/03/22
5	Definición de Objetivos	vie 18/03/22	mié 23/03/22
6	Elaboración del índice	mar 22/03/22	mié 23/03/22
7	Estado del arte	jue 24/03/22	dom 03/04/22
8	**Fase de Análisis y Planificación**	**dom 03/04/22**	**sáb 23/04/22**
17	**Fase de Diseño**	**mar 26/04/22**	**vie 20/05/22**
25	**Fase Implementacion**	**sáb 21/05/22**	**dom 12/06/22**
31	**Fase Final**	**sáb 21/05/22**	**jue 23/06/22**

Figura 2.6: Gantt: Fase Inicial

Id	Nombre de tarea	Comienzo	Fin
1	Comienzo Proyecto	lun 14/03/22	lun 14/03/22
2	**Fase Inicial**	**lun 14/03/22**	**dom 03/04/22**
8	**Fase de Análisis y Planificación**	**dom 03/04/22**	**sáb 23/04/22**
9	Análisis de Requisitos	dom 03/04/22	lun 04/04/22
10	Preparación de Calendario y Estimación temporal	mar 05/04/22	jue 07/04/22
11	Planificación de presupuesto	jue 07/04/22	jue 07/04/22
12	Planificación y Gestión de Riesgos	vie 08/04/22	dom 10/04/22
13	Estudio sobre las Herramientas a utilizar	dom 10/04/22	vie 15/04/22
14	Análisis de los datos disponibles	vie 15/04/22	sáb 16/04/22
15	Aprendizaje NIFI - Realizacion de ejemplos	sáb 16/04/22	sáb 23/04/22
16	Repaso Power BI - Realizacion de ejemplos	sáb 16/04/22	mar 19/04/22
17	**Fase de Diseño**	**mar 26/04/22**	**vie 20/05/22**
25	**Fase Implementacion**	**sáb 21/05/22**	**dom 12/06/22**
31	**Fase Final**	**sáb 21/05/22**	**jue 23/06/22**

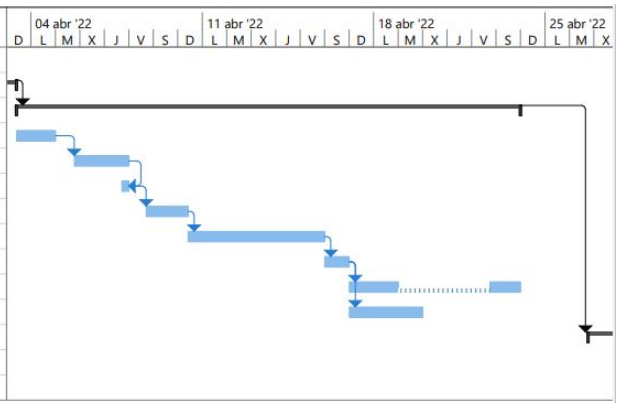


Figura 2.7: Fase de Análisis y Planificación

Id	Nombre de tarea	Comienzo	Fin
1	Comienzo Proyecto	lun 14/03/22	lun 14/03/22
2	**Fase Inicial**	**lun 14/03/22**	**dom 03/04/22**
8	**Fase de Análisis y Planificación**	**dom 03/04/22**	**sáb 23/04/22**
17	**Fase de Diseño**	**mar 26/04/22**	**vie 20/05/22**
18	Elaboración de Caso de Uso	mar 26/04/22	jue 28/04/22
19	Elaboración del Modelo Dimensional	lun 02/05/22	jue 05/05/22
20	Diseño de la BBDD	lun 09/05/22	mar 10/05/22
21	Estructuración de Datos y Directorios	lun 09/05/22	mar 10/05/22
22	Diseño Ingesta	mié 11/05/22	sáb 14/05/22
23	Diseño ETL - Proceso de cálculo	sáb 14/05/22	lun 16/05/22
24	Diseño Visualización	mar 17/05/22	vie 20/05/22
25	**Fase Implementacion**	**sáb 21/05/22**	**dom 12/06/22**
31	**Fase Final**	**sáb 21/05/22**	**jue 23/06/22**

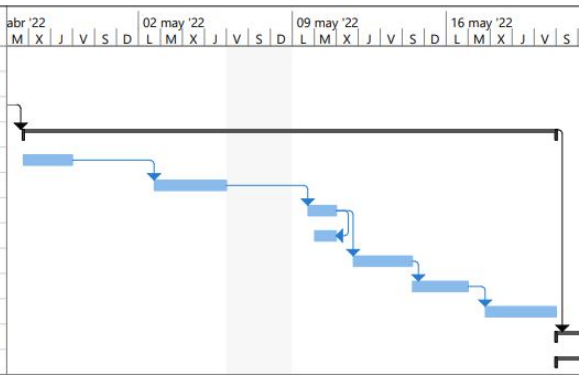


Figura 2.8: Gantt: Fase de Diseño

Id	Nombre de tarea	Comienzo	Fin
1	Comienzo Proyecto	lun 14/03/22	lun 14/03/22
2	**Fase Inicial**	**lun 14/03/22**	**dom 03/04/22**
8	**Fase de Análisis y Planificación**	**dom 03/04/22**	**sáb 23/04/22**
17	**Fase de Diseño**	**mar 26/04/22**	**vie 20/05/22**
25	**Fase Implementacion**	**sáb 21/05/22**	**dom 12/06/22**
26	Creacion de entornos virtuales	sáb 21/05/22	sáb 21/05/22
27	Creación de tablas de la BBDD	sáb 21/05/22	sáb 21/05/22
28	Implementación de la Ingesta	sáb 21/05/22	vie 27/05/22
29	Implementación de ETL - Proceso de cálculo	sáb 28/05/22	lun 06/06/22
30	Implementación de la Visualización	lun 06/06/22	dom 12/06/22
31	**Fase Final**	**sáb 21/05/22**	**jue 23/06/22**

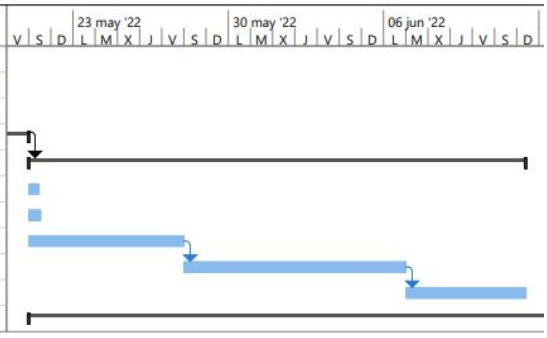


Figura 2.9: Gantt: Fase de Implementación

Id	Nombre de tarea	Comienzo	Fin
1	Comienzo Proyecto	lun 14/03/22	lun 14/03/22
2	**Fase Inicial**	**lun 14/03/22**	**dom 03/04/22**
8	**Fase de Análisis y Planificación**	**dom 03/04/22**	**sáb 23/04/22**
17	**Fase de Diseño**	**mar 26/04/22**	**vie 20/05/22**
25	**Fase Implementacion**	**sáb 21/05/22**	**dom 12/06/22**
31	**Fase Final**	**sáb 21/05/22**	**jue 23/06/22**
32	Inicio de la fase de pruebas	sáb 21/05/22	sáb 21/05/22
33	Documentacion de pruebas	lun 20/06/22	mar 21/06/22
34	Fin de la fase de pruebas	mar 21/06/22	mar 21/06/22
35	Conclusiones y Trabajo futuro	mié 22/06/22	mié 22/06/22
36	Repaso final	jue 23/06/22	jue 23/06/22

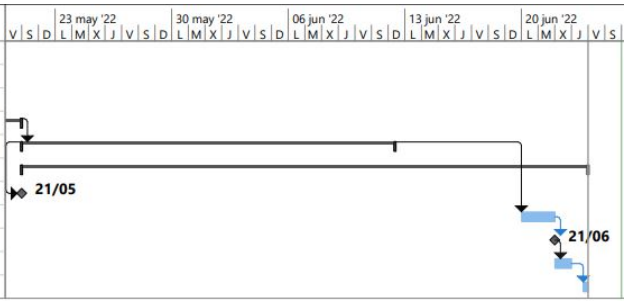


Figura 2.10: Gantt: Fase Final

2.3. Presupuesto Económico.

En esta sección del documento se recogen los costes asociados a la elaboración del proyecto, de acuerdo a la planificación del apartado 2.2. Debe tenerse en cuenta que, dado el contexto académico en el que se realiza y al carecer de unas necesidades económicas reales, se trata de una simulación y su valor real será

mucho menor del indicado. Dicho Presupuesto esta dividido en tres partes, presupuesto de mano de obra, presupuesto hardware y presupuesto software.

En primer lugar, ha de tenerse en cuenta los costes generados por los recursos humanos necesarios para desarrollar el proyecto. Para poder obtener una estimación del salario medio de un Ingeniero Informático se han consultado distintos portales de empleo donde sus usuarios comparten sus salarios y opiniones. La cantidad que utilizaremos ha sido consultada a fecha 12 de mayo de 2022 establece un salario medio de 26.500€ [12]/26.326€ [17] anuales para un Ingeniero Informático con cierta experiencia y en concreto 22.500€ [17] brutos anuales para un recién titulado. Si además consideramos que una jornada laboral es de 40 horas semanales y un año cuenta con 52 semanas, tendremos que dividir el salario bruto anual entre 2.080 horas de trabajo anual (52x40), obteniendo un salario por hora de 10,82€. Al gasto del salario deben añadirse los gastos por cotizaciones a la Seguridad Social, que debe pagar la empresa para cubrir parte de los gastos sociales del empleado. La cifra aproximada es del 30 % del sueldo bruto, aunque a continuación vemos un desglose del este porcentaje real [16]:

- Cotización por contingencias comunes: 23,60 % del salario bruto.
- Cotización por formación: 0,60 %.
- Cotización por desempleo: 6,70 % en contratos temporales.
- Cotización por accidentes de trabajo y enfermedades profesionales. Grupo 62 [16]: Programación, consultoría y otras actividades relacionadas con la informática. 1,50 %
- Cotización al Fondo de Garantía Salarial (FOGASA): 0,20 %.

El porcentaje total será del 32,6 % mensual sobre el salario bruto. El trabajador recibirá un total de 2.956,73€ en un periodo de 4 meses, por lo que recibiría de media 729,18€/mes. En definitiva, el gasto por contingencias a la seguridad social por parte de la empresa seria de 237,71€/mes. En la Tabla 2.7 se muestran todos estos gastos.

Concepto	**Coste**	**Duración del Proyecto**	**Salario Total**
Salario Ingeniero Informático	9,85€/Hora	300 horas	2.956,73€
Gastos Seguridad social	237,71€/mes	4 meses	950,85€
		Total Presupuesto Personal	3.907,58€

Tabla 2.7: Coste de Personal

En cuanto al presupuesto software, van a utilizarse las aplicaciones y herramientas descritas en el apartado 3. La mayoría de estas son de software libre por lo que no han supuesto ningún gasto añadido. Para calcular el coste del software se han tenido en cuenta la duración en meses del proyecto, desde el inicio en Marzo hasta su finalización en Junio, siendo un total de 4 meses de uso de estas herramientas.

Nombre Software	**Tipo Paquete Adquirido**	**Precio Mensual**	**Precio Total**
Microsoft Office 365	Empresa Estándar	10,50 €	42,00 €
MS Project	Plan 3	25,30 €	101,20 €
Microsoft Power BI	Power BI Desktop	Gratuito	0,00 €
Heidi SQL	Software libre	Gratuito	0,00 €
Maria DB	Software libre	Gratuito	0,00 €
Apache NIFI	Software libre	Gratuito	0,00 €
		Total Presupuesto software	143,20€

Tabla 2.8: Presupuesto Software

En la imagen 2.11 se muestran los distintos paquetes Microsoft Office 365 ofrecidos para empresas y las aplicaciones y servicios incluidos. En ninguno de ellos se incluye el software MS Project, utilizado para la planificación del proyecto, por lo que deberá adquirirse por separado, en la imagen 2.12 se muestran los distintos paquetes para este otro software.

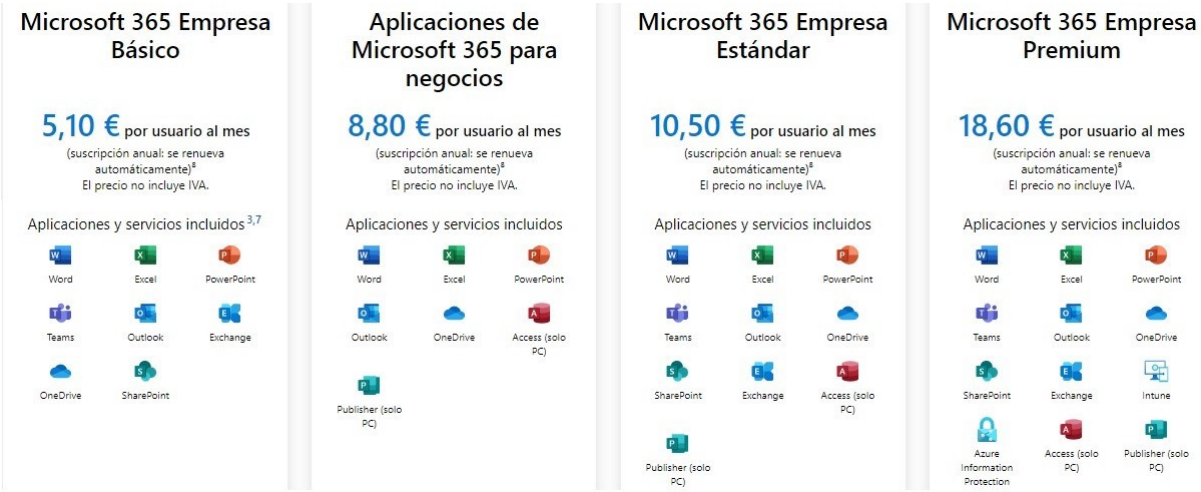


Figura 2.11: Paquetes Microsoft Office 365 y aplicaciones incluidas

Figura 2.12: Paquetes MS Project

En cuanto al presupuesto hardware, se han añadido los equipos informáticos necesarios para que el estudiante pueda desarrollar el proyecto. Estos gastos se imputan como costes por medio de amortización, cuyo tiempo mínimo es de 4 años [15] [14] . Puesto que la duración del proyecto serán 4 meses, se calcularán los costes del presupuesto para dicho periodo, los cuales pueden verse en la tabla 2.9.

Cantidad	**Artículo**	**Precio**	**Tiempo de Amortización**	**Cuota Amortización Mensual**	**Tiempo de Uso**	**Amortización Total**
1	Ordenador Personal	479,00 €	4 años	9,98€	4 meses	39,92 €
1	Monitor	119,99 €	4 años	2,50€	4 meses	10 €
Total Presupuesto Hardware		**598,99€**	Total Presupuesto Hardware Amortizado			**49,92€**

Tabla 2.9: Presupuesto Hardware

2.3.1. Presupuesto Final

Finalmente, a modo de resumen, en la siguiente tabla 2.10 se muestra los gastos mencionados anteriormente y el monto total del presupuesto de este proyecto.

Concepto	**Gasto Total**
Presupuesto de Personal	3.907,58€
Presupuesto Software	143,20€
Presupuesto Hardware	49,92€
Presupuesto FINAL	**4100,68€**

Tabla 2.10: Presupuesto Final

Capítulo 3

Marco Tecnológico

En este apartado se van a describir las herramientas utilizadas durante el TFG. La primera de ellas, utilizada en el apartado anterior, es ***MS Project***, un software de gestión de proyectos que ayudará a realizar la planificación de este proyecto y realizar un seguimiento de las tareas y sus fechas de entrega estimadas.

Para crear y administrar las bases de datos y tablas de este proyecto utilizaremos Maria DB y la herramienta ***Heidi SQL***. Comenzando con el proceso ETL, utilizaremos ***Apache NIFI*** para realizar la Ingesta inicial de datos.

Como lenguaje de programación utilizado para realizar las transformaciones necesarias para purgar y dar calidad a los datos, realizar el cálculo de los SLAs que definiremos, conectar con las BBDD y exportar los resultados de los datos, se ha escogido ***Python***.

En ocasiones, los analistas y científicos de datos utilizan múltiples versiones para los distintos paquetes y crean distintos entornos para alojar cada uno de ellos. En este proyecto crearemos un entorno para cada uno de los contratos que tenga el proveedor con los distintos clientes y utilizaremos el software ***Anaconda Navigator*** para ello.

Por último, utilizaremos ***Power BI Desktop*** para el análisis y la visualización de los resultados obtenidos.

3.0.1. MS Project

En cualquier proyecto es fundamental tener un software de gestión de proyectos que nos permita desarrollar estos de manera eficiente y con el que podamos trabajar de forma colaborativa y con información actualizada. Un flujo de trabajo mal definido y una mala gestión puede poner en riesgo el éxito de cualquier proyecto y estas herramientas nos permiten evitar retrasos y asegurar que los proyectos se entreguen a tiempo y dentro de lo previsto.

Microsoft Project es una a herramienta software de administración de proyectos diseñada por Microsoft y utilizada por administradores, colaborades y jefes de proyectos. Su uso es muy sencillo e intuitivo, con ella podemos realizar un seguimiento de los proyectos de forma colaborativa y generar reportes de avance, así como plasmar las necesidades de un proyecto, como duración, costos, entregables, actividades, recursos, calendarios, etc. Con esta herramienta podremos calcular costes y controlar los recursos asignados para cada persona y tarea y así evitar un uso de material innecesario.

Microsoft Project favorece la productividad entre el equipo de trabajo y reduce el tiempo de calendarizar y organizar tareas [20]. Con él podemos gestionar la asignación de tareas y recursos, conocer las tareas más importantes y esenciales para nuestro proyecto y tener una visión global de su estado en cualquier momento. Dispone de diagramas de Gantt y menús desplegables que simplifican todo el proceso de planificación de proyectos.

3.0.2. Heidi SQL

Es una aplicación de escritorio de código abierto para la administración de bases de datos MySQL y Microsoft SQL, ligera y con una interfaz muy amigable. Tiene mayor potencia que otras herramientas de gestión de BBDD como PHPMyAdmin ya que a diferencia de esta, Heidi SQL es capaz de exportar e importar bases de datos de gran tamaño. Además permite importar y exportar datos desde o hacia distintas fuentes o bases de datos y con distintos formatos, CSV, XML, SQL, etc.

Esta herramienta permite visualizar y modificar datos alojados en bases de datos MariaDB, MySQL, SQLite, etc. así como crear y editar tablas, vistas, procedimientos, triggers y eventos programados de forma rápida. Heidi SQL permite conectar con varios servidores a la vez en una única ventana y guardar múltiples sesiones con conexiones y credenciales.

3.0.3. Apache NIFI

NIFI es un proyecto de software de Apache diseñado para automatizar el flujo de datos entre diferentes sistemas de software. Se trata de un sistema fácil de usar, potente y fiable para procesar y distribuir datos. Apache Nifi se basa en una tecnología anteriormente llamada “Niagara Files” [4], desarrollada y utilizada por la NSA (Agencia de Seguridad Nacional de EEUU),

Aunque se puede considerar una herramienta ETL, ya que permite extraer, transformar y cargar datos, no está optimizado para realizar transformaciones de datos complejas. Aun así ofrece grandes ventajas dentro del Big Data, como herramienta de automatización de ingesta de datos y para la realización de transformaciones y limpiezas simples.

Esta herramienta se basa en un modelo de programación basado en el flujo de datos. Este término se utiliza en distintos contextos, pero en este caso hace referencia al flujo automatizado y gestionado de información entre sistemas [4]. Nifi es un programa desarrollado en Java, que se ejecuta dentro de una JVM (Máquina Virtual de Java).

3.0.4. Python

Python es un lenguaje de programación muy flexible que con los años ha ganado cada vez más peso en el mundo del Big Data. Aunque existen otros lenguajes para realizar análisis de datos y tareas científicas y numéricas, como R, Scala, Matlab o Julia, Python ha logrado imponerse sobre ellos, entre otras razones gracias a su facilidad de aprendizaje. Además, cuenta con una amplia comunidad de desarrollo, donde resulta muy sencillo colaborar o consultar cualquier problema.

Python proporciona todas las herramientas necesarias para realizar la recolección, limpieza, exploración, modelado y visualización de datos. Estas pueden ser tareas propias de cualquier proceso en el que esté implicado un analista de datos, ya que cuenta con librerías dedicadas para realizar estos pasos, como Pandas, Numpy, Matplotlib, SciPy, etc. Pero para este proyecto únicamente utilizaremos las dos primeras.

- **Pandas**: es una de las más versátiles y robustas, esencial para manipular Dataframes con Python. Es una herramienta muy útil para crear procesos ETL (Extract-Transform-Load) ya que permite la manipulación de datos de manera fácil e intuitiva. Además, ofrece soporte para formatos de datos comunes (archivos CSV, Excel, lectura de bases de datos SQL...).
- Librería **Numpy** : especializada en el cálculo numérico y el análisis de datos, se trata de la librería por excelencia para aplicar informática científica. Este paquete contiene funciones matemáticas de alto nivel para operar con matrices, incluso, de más de dos dimensiones.

3.0.5. Anaconda Navigator

Anaconda es una Suite de código abierto que abarca una serie de aplicaciones, librerías y conceptos diseñados para el desarrollo de la Ciencia de datos con Python [5]. Se trata de una de las mejores distribuciones de Python que da soporte al desarrollo científico, tanto analítico como gráfico y que funciona como un gestor de entornos y paquetes. Cuenta con una gran colección de paquetes de código abierto (más de 720) y gran cantidad de funcionalidades que permiten desarrollar proyectos de forma eficaz, rápida y sencilla [3].

Anaconda Distribution se agrupa en 4 sectores o soluciones tecnológicas que se instalan de forma automática, *Anaconda Navigator*, *Anaconda Project*, librerías de *Ciencia de datos* y *Conda*. **Anaconda Navigator** es una interfaz gráfica de usuario (GUI) muy sencilla pero de gran potencial, con la que podemos gestionar las herramientas instaladas, así como instalar y administrar paquetes, dependencias y entornos de manera sencilla, sin necesidad de utilizar la terminal.

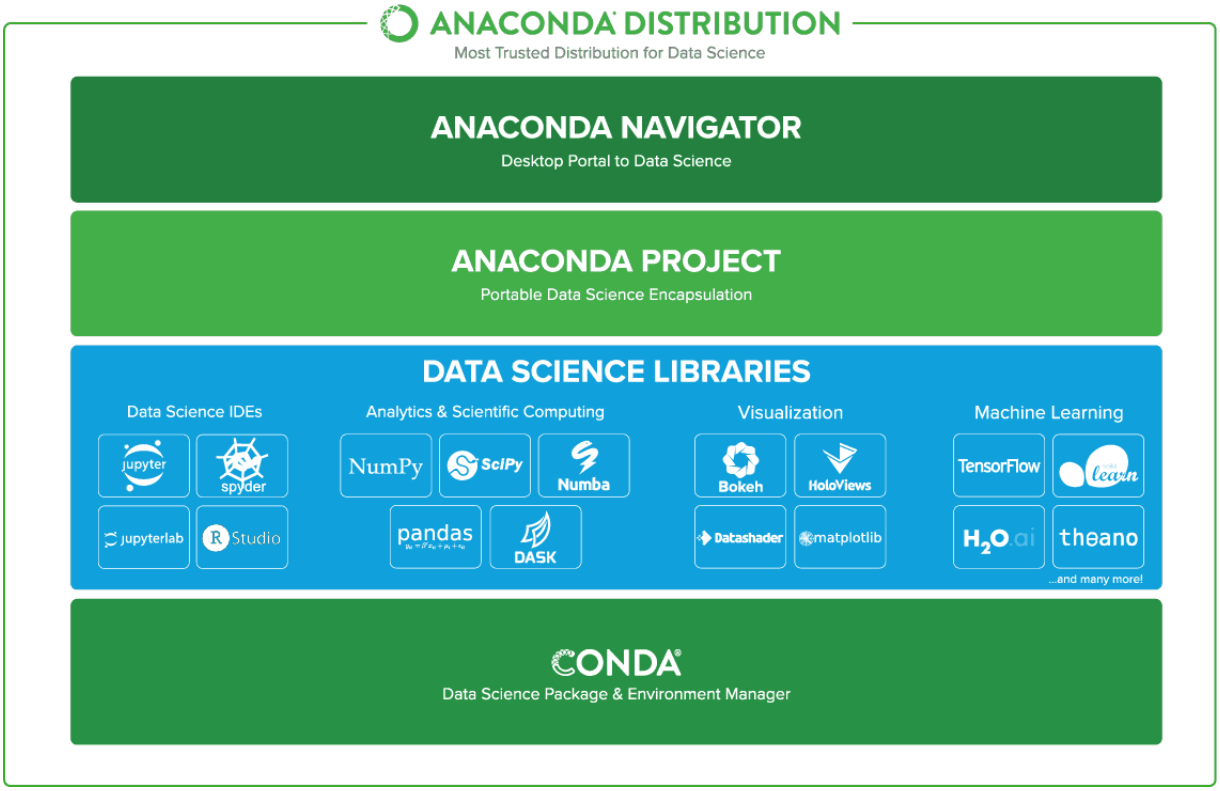


Figura 3.1: Anaconda Distributuion

3.0.6. Microsoft Power BI

Microsoft Power BI es una suite de herramientas de análisis empresarial que permite analizar y visualizar datos en una sola vista y ayuda a las organizaciones a entender las tendencias de su negocio. Es un sistema

predictivo, inteligente y de gran apoyo, capaz de traducir los datos (simples o complejos) en gráficas, paneles o informes [21]. Se utiliza principalmente para crear cuadros de mando que contribuyen a la toma de decisiones empresariales (tácticas y estratégicas), los cuales estarán adaptados a las necesidades e infraestructura de cada departamento de la organización.

Esta suite se basa en Microsoft Excel y está ideada para aquellas organizaciones que quieren obtener mayor capacidad para generar informes y mayor capacidad analítica de sus datos. Power BI es un término general y puede referirse a una aplicación de escritorio (Power BI Desktop), un servicio en línea SaaS (Power BI Service), o aplicaciones móviles. Para este proyecto utilizaremos la aplicación gratuita de escritorio Power BI Desktop.

Con esta herramienta podemos obtener información de valor tras procesar datos desde distintas fuentes de datos, para analizarla y visualizarla. Power BI permite la conexión a distintos orígenes de datos desde hojas de cálculo básicas de Excel, ficheros csv, hasta bases de datos, con los que puede crear informes en tiempo real y en cualquier dispositivo. Al estar alojado en la nube es posible acceder fácilmente a los datos desde cualquier lugar y compartir los análisis en tiempo real con diferentes usuarios de la misma organización, sólo es necesario tener una conexión a internet.

Capítulo 4

Análisis

En este capítulo se van a describir los requisitos del sistema, tanto funcionales como no funcionales y se incluirán los Casos de uso (CU) del sistema, en este caso un único CU.

4.1. Requisitos

Los Requisitos funcionales describen la funcionalidad que tendrá un sistema o los servicios que prestará, mientras que los requisitos no funcionales describen restricciones a las que debe ajustarse nuestro sistema, pueden clasificarse en requerimientos de producto, organizacionales y externos. Los primeros se refieren a propiedades del sistema como rendimiento, usabilidad, seguridad, disponibilidad, etc. Los organizacionales se refieren a políticas internas de la organización, requisitos de implementación como lenguajes de programación o métodos a utilizar, etc. Por último los Externos son requisitos legales y éticos que deben seguirse.

4.1.1. Requisitos Funcionales

RF01 El sistema deberá poder realizar todo el proceso de ingesta y procesamiento de datos de manera automatizada, sin intervención de ningún usuario.

- **RF01.1** El sistema deberá realizar un borrado de datos antes de cargar nuevos en la BBDD.
- **RF01.2** El sistema deberá procesar tickets de red que se encuentren cerrados o solucionados.
- **RF01.3** El sistema deberá completar la información sobre tickets que no esté registrada.
- **RF01.4** El sistema no deberá procesar tickets que no cumplan las validaciones de calidad del dato.
- **RF01.5** El sistema deberá calcular tiempos de resolución de cada ticket.
- **RF01.6** El sistema deberá comprobar si cada ticket cumple los plazos de resolución acordados.
- **RF01.7** El sistema deberá calcular el porcentaje de cumplimiento de cada KPI definido.
- **RF01.8** El sistema deberá obtener información que sirva para visualizar el cumplimiento de los acuerdos de servicio.

RF02 El sistema deberá poder permitir realizar ejecuciones manuales del proceso ETL (encargado de procesar los datos).

RF03 El sistema deberá anonimizar los datos.

RF04 El sistema deberá permitir realizar ejecuciones con distintas perioricidades, Mensual y Semanal.

RF05 El sistema deberá cargar datos en formato csv, delimitado por ';'.

RF06 El sistema deberá almacenar y actualizar la información en la BBDD.

4.2. Requisitos NO funcionales

RNF01 El sistema deberá utilizar las siguientes versiones de librerias:

- python 3.6.9
- openpyxl 2.4.8
- pandas 1.1.1
- sqlalchemy 1.3.16
- pymysql 0.9.3
- numpy 1.19.1

RNF02 La BBDD utilizada debe ser Maria DB.

RNF02 El sistema deberá almacenar información sobre el estado de las ejecuciones.

RNF03 El sistema deberá almacenar ficheros log en texto plano.

RNF04 El sistema deberá almacenar los ficheros ingestados en formato csv, delimitado por ';'

RNF05 El sistema deberá almacenar los ficheros de resultados en formato Excel.

RNF06 El sistema deberá garantizar la detección y recuperación de errores.

RNF07 El sistema deberá funcionar sobre Windows.

RNF08 El sistema se conectará a BBDDs externas para ampliar información.

4.3. Casos de Uso

Los casos de uso definen una secuencia de acciones realizado por el sistema para llevar a cabo una de sus funcionalidades. Gracias a ellos podemos describir el comportamiento de nuestro sistema. A continuación describiremos el único caso de uso generado, para describir la ejecución manual del proceso ETL de cálculo.

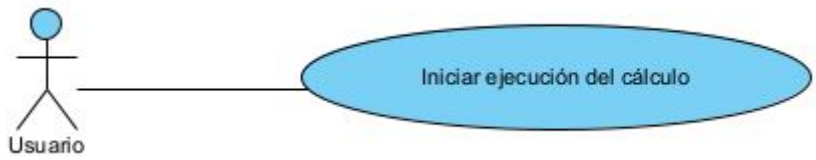


Figura 4.1: Diagrama de Caso de Uso

CU-01	Iniciar Ejecución del cálculo
Descripción	El sistema deberá permitir realizar ejecuciones manuales del proceso ETL.
Actor	Administrador del Sistema
Descripción	El usuario inicia una ejecución automática del proceso ETL
Precondición	El usuario debe haber accedido al gestor de BBDD y poseer permisos de root sobre la BBDD 'bbdd_cumplimientos_sla'
Secuencia	
Paso	**Acción**
1	El Actor introduce una nueva tupla en la tabla ctrl_estado_ejecucion de la BBDD
2	El Sistema refleja la tupla introducida en la tabla y le asigna un identificador.
3	El Actor accede al directorio `/mnt/c/My_TFG/src/app/tools/slas_C001/bin` desde WSL (Ubuntu).
4	El Actor ejecuta el script *calculo_slas_c001.sh* (Figura 6.2.1) con el identificador obtenido en el paso 2.
Postcondición	El sistema ha ejecutado el script de forma correcta y ha introducido en la BBDD los resultados de la ejecución. El Sistema ha volcado los resultados en los ficheros .xlsx correspondientes.

Tabla 4.1: Secuencia Caso de Uso

4.4. Diagramas UML

4.4.1. Diagrama de Actividad

A continuación, en la figura 4.2 se ha descrito un diagrama de actividad sencillo que ilustra el flujo de trabajo que realiza el proceso de ingesta de datos a través de nifi. El procedimeinto tiene una entrada de datos que consiste en un fichero csv, este lo leerá y a continuación realizará una consulta SQL para conocer la tabla destino donde cargar los datos. A partir de los datos leídos del fichero y el resultado de la consulta, se cargará a información en la tabla correspontiente y finalizará la secuencia.

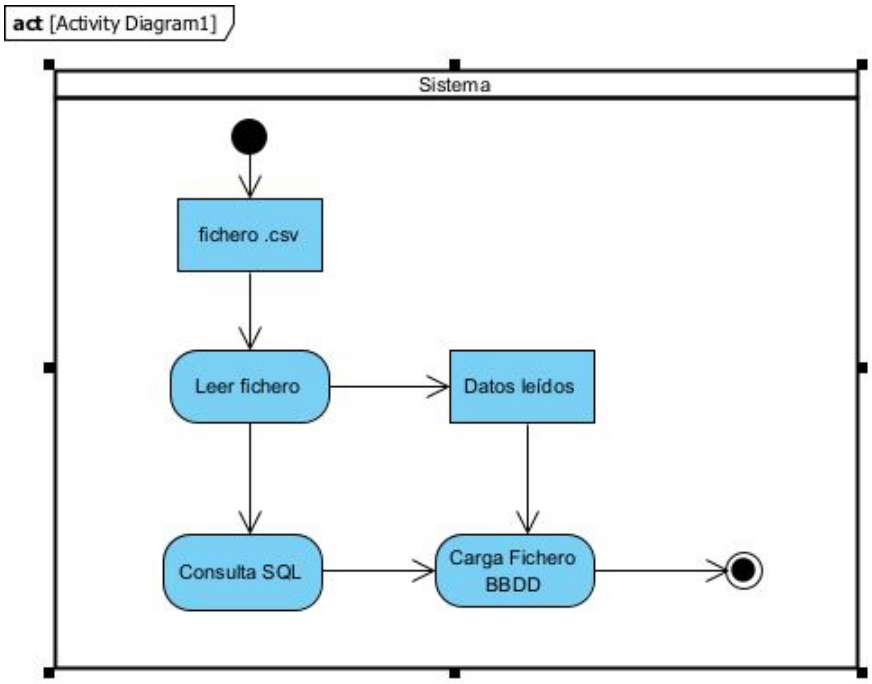


Figura 4.2: Diagrama de Actividad

4.4.2. Diagrama de Secuencia CU-01

El diagrama de secuencias ilustrado en la figura 4.3 corresponde a la realización del Caso de Uso 1. Este diagrama nos permite especificar el comportamiento que tendrá el sistema cuando el usuario interactúa con él para iniciar el procesamiento de datos de forma manual.

Para comenzar, el usuario realiza una llamada al script *calculo_slas_c001.sh* con un identificador de ejecución, que deberá existir previamente en la tabla ctrl_estado_ejecucion de la BBDD. A continuación el script creará el fichero log donde se almacenará la información y errores que surgan durante la ejecución y realizará una llamada al proceso *inicio_ejecucion.py* con la ruta al fichero log y el id de ejecución. Este proceso se encarga de comprobar que los datos de la ejecución son válidos y que existan tickets en la BBDD para poder comenzar la ejecución.

El proceso *inicio_ejecucion.py* realiza una consulta sql a la tabla *ctrl_estado_ejecucion* para conocer los metadatos asociados a la ejecución que queremos realizar (contrato, sla, tipo de ejecución, periodo de cálculo...). A continuación realizará una nueva consulta sobre esta tabla para actualizar el resultado de la ejecución, indicando que ha iniciado ('WORK IN PROGRESS'). El proceso comprobará que los datos introducidos en la tabla sean correctos, si no es así, actualizará el estado de la ejecución indicando que esta ha fallado ('KO'), devolverá el control al script que finalizará la ejecución.

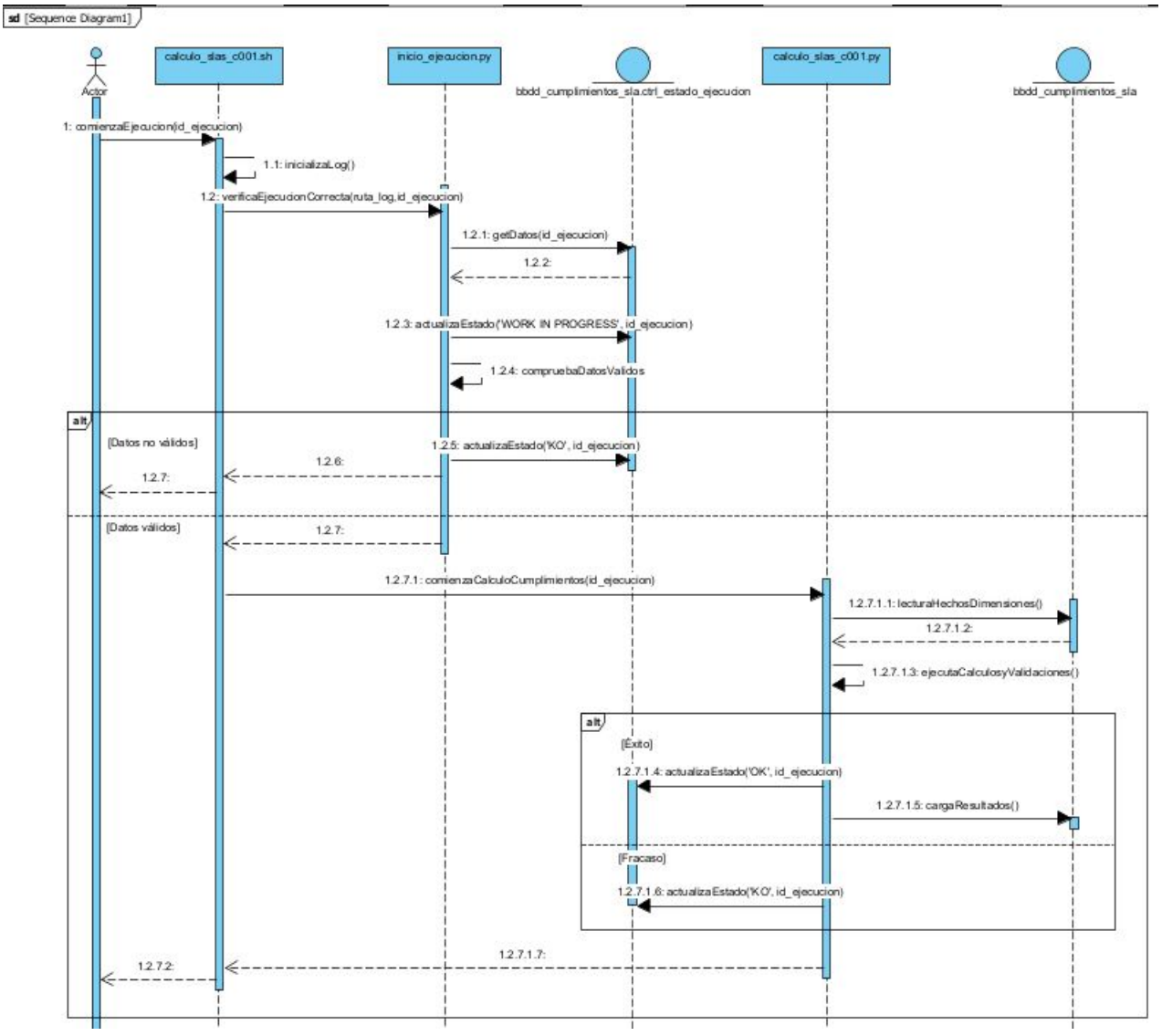


Figura 4.3: Diagrama de Secuencia

En caso de que los datos sean correctos, devolverá el control al script, el cual realizará una nueva llamada al proceso *calculo_slas_c001.py*. Este realizará distintas consultas a la BBDD para obtener los datos de las tablas de dimensiones y hechos que permitirán realizar todos los cálculos, transformaciones, uniones necesarias. si todo este procesamiento es correcto, se realiza una consulta sql a la tabla *ctrl_estado_ejecucion* para actualizar el resultado de la ejecución, indicando que ha finalizado con écito ('OK') y se cargarán los resultados obtenidos en la BBDD. En caso de que se produzca algún error, se actualizará el estado de la ejecucion con un 'KO'. En cualquiera de los dos supuestos, la ejecución finalizará y devolverá el control al script que terminará la secuencia.

Capítulo 5

Diseño

5.1. Arquitectura Lógica

La arquitectura lógica permite visualizar como interactúan y se relacionan los distintos componentes del sistema construido. En la siguiente figura se pueden ver las distintas fases por las que pasan los datos, desde el origen hasta la carga final en la BBDD y los directorios locales.

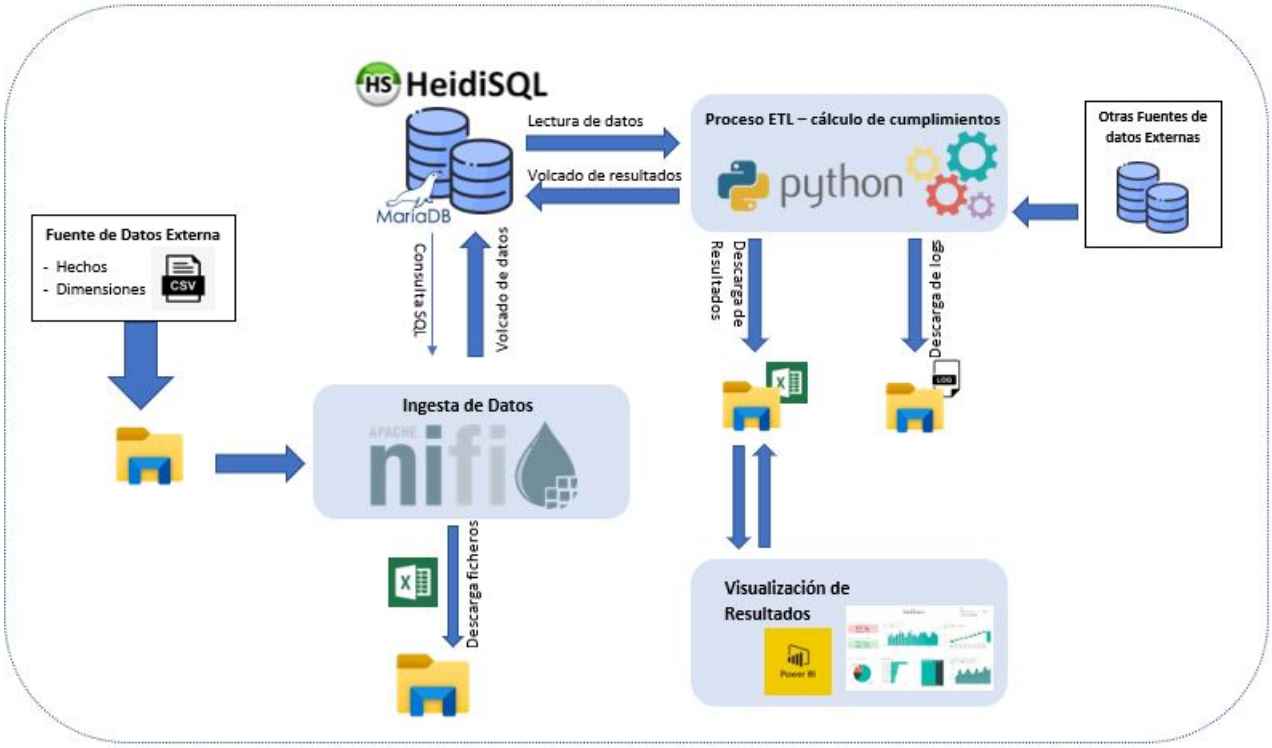


Figura 5.1: Arquitectura Lógica/Funcional

5.2. Tipos de datos utilizados

Antes de pasar a explicar las tablas y BBDD utilizadas para desarrollar este proyecto, se van a describir los tipos de ficheros de los que disponemos para ello. En primer lugar tenemos ficheros de entrada, Hechos y Dimensiones, proporcionados por la empresa NTT Data de acuerdo a unos estándares preestablecidos con la empresa de Telecomunicaciones. Estos ficheros, que tendrán formato CSV, se volcarán en la BBDD de manera total borrando los datos antiguos y cargando los nuevos. Por otro lado, tendremos ficheros de salida, logs y resultados, que se almacenarán localmente para su posterior análisis, en texto plano y excel

respectivamente.

1. **Hechos (Facts):** Contiene la información que queremos medir y analizar, información sobre cada ticket como su identificador, fecha de apertura, severidad, elemento de la red al que corresponde, tipo de tecnología, tipo de ticket, etc. Estos datos se volcarán a una tabla de la BBDD, identificada con el prefijo '*fact*'.

2. **Dimensiones:** son datos más simples que permiten ampliar la información de los hechos y ayudar a procesar y analizarlos más eficientemente. Las dimensiones se cargarán en distintas tablas de la BBDD con el prefijo '*dim*'. No existe un límite a la cantidad de dimensiones que pueden manejarse, la mayoría de ellas han sido proporcionadas por NTT Data y otras se han creado para cumplir algún requisito, como la anonimización de tickets. Los ficheros de dimensiones que manejaremos son los siguientes:

   - Dimensiones Provincias: Se trata de un conjunto de tres dimensiones que permiten conocer la zona geográfica (denominada site) a la que pertenece cada ticket a partir del campo de provincia (desnormalizada).
   - Dimensión severidad-prioridad: Contiene una asociación entre niveles de prioridad, índices de severidad y tecnología asociada a cada ticket. Esta dimensión nos permitirá asociar un nivel de prioridad a cada ticket, a partir de los campos de severidad y tecnología.
   - Dimensión cumplir prioridad-site Contrato c001: Esta dimensión contiene los tiempos de resolución y restauración que deberán cumplir los tickets para considerar que han cumplido el acuerdo. Estos tiempos variarán dependiendo del nivel de prioridad y el site asociado a cada ticket.
   - Dimensión sla cumplir Contrato c001: Contiene detalles sobre los KPIs que deben cumplirse para el acuerdo de servicio asociado al Contrato C001. Estos detalles son: identificador, descripción del KPI a cumplir, descripción sobre el % de cumplimiento y valor del porcentaje.
   - Dimensión perfil técnicos (Creada): Contiene una relación entre códigos alfanuméricos de 8 caracteres e identificadores personales que distinguen a cada uno de los técnicos/agentes que interactúan con los tickets. Esta dimensión se ha creado para ofrecer mayor anonimato a la hora de publicar resultados ya que el id original contenía información personal de estos técnicos.
   - Dimensión cifrado (Creada): Contiene una relación entre cada letra del abecedario y un valor numérico asociado. Esta dimensión sirve para realizar un cifrado alfanumérico sobre los identificadores de los tickets, para no publicar los originales.

3. **Logs:** Ficheros que contendrán mensajes de error e información relevante sobre las ejecuciones. Serán útiles para identificar fallos en la implementación de los procesos o en sus ejecuciones.

4. **Resultados (Outputs):** Almacenarán los resultados del cálculo de cumplimiento de SLAS, incluyendo información de métricas para su posterior análisis. Los resultados se cargarán en tablas de la BBDD con el prefijo '*out*'. Estos son generados tras la ejecución del proceso y a su vez se clasifican en 3 tipos:

   - *Resultados:* Contiene aquellos tickets analizados durante la ejecución, que cumplen con las validaciones de calidad de dato.
   - *Descartados:* Contiene los tickets analizados que no cumplen las validaciones de calidad del dato y se almacenan en un fichero diferente.
   - *Resumen:* Contiene tuplas que resumen los resultados obtenidos tras cada ejecución, incluyendo porcentajes de cumplimiento.

5.3. Modelo Dimensional

El Modelado Dimensional es un conjunto de técnicas y conceptos utilizados para el diseño lógico de almacenes de datos. Este ofrece un alto rendimiento a la hora de acceder a los datos y está ligado a la Inteligencia de Negocio, ya que que el diseño está orientado a apoyar las consultas de los usuarios finales. El objetivo es que estos puedan acceder de forma rápida e intuitiva a la información que necesiten. Este modelo utiliza conceptos del modelo entidad/relación pero se distinguen en lo siguiente:

- No hace distinción entre relaciones, sino entre hechos y dimensiones.
- El rendimiento de las consultas en un modelo dimensional es mayor, ya que utiliza menos uniones para realizar consultas.
- El modelo ER tiene datos normalizados, mientras que los datos del modelo dimensional están desnormalizados.
- El modelo ER mantiene datos transaccionales actuales mientras que el Dimensional mantiene el resumen de transacciones actuales e históricos.

El diseño de la BBDD viene preestablecido por NTT Data y la empresa de Telecomunicaciones y es el que se muestra en la figura 5.2. Algunas de las entidades de dimensiones representadas no están directamente relacionadas con la entidad de hechos, ya que no existen atributos que puedan relacionarse directamente. Parte del proceso ETL será aplicar operaciones lógicas, transformaciones y uniones entre tablas para poder obtener estos atributos y relacionar las dimensiones con la tabla de hechos.

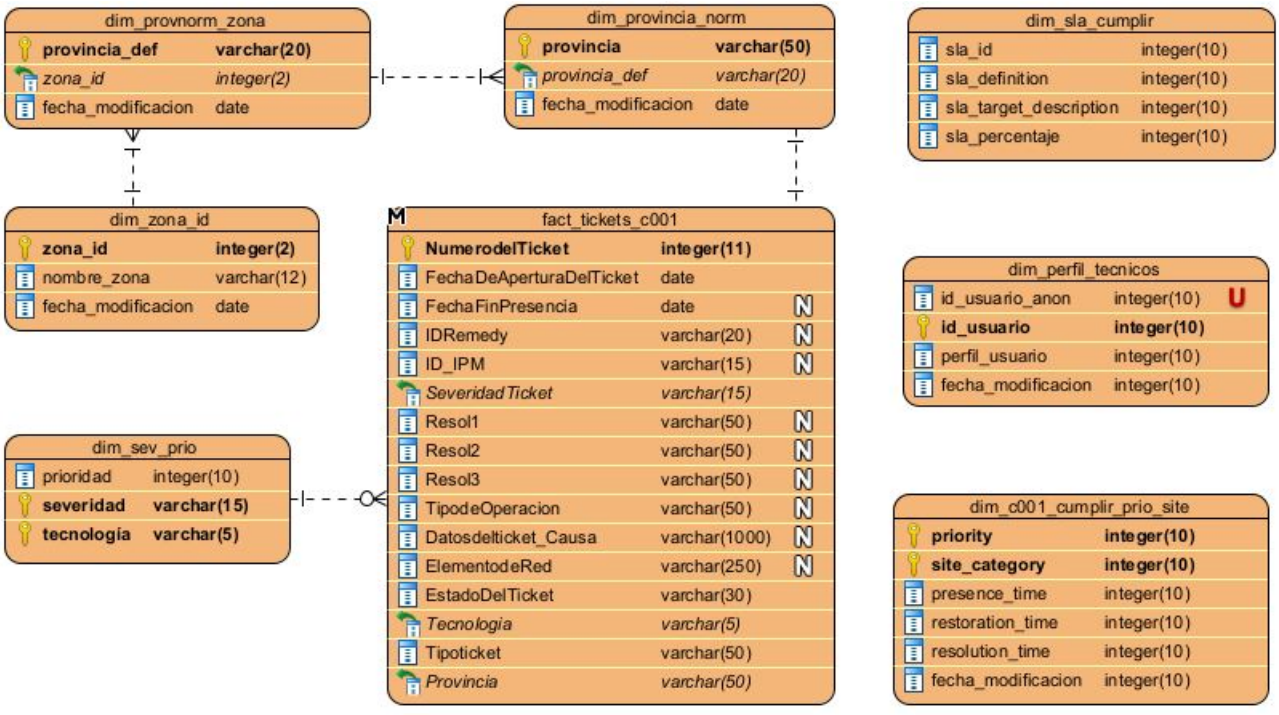


Figura 5.2: Modelo dimensional

5.4. Descripción de las tablas y BBDDs utilizadas

Como se ha visto en el apartado 3, la herramienta utilizada para crear y administrar las bases de datos necesarias para este proyecto ha sido Heidi SQL. En total, se han utilizado 3 Bases de Datos distintas, de las cuales únicamente se creado y gestionado una (*bbdd_cumplimientos_sla*). El resto son BBDDs creadas y administradas por la empresa de Telecomunicaciones a las cuales accederemos de forma remota. Estas serán de ayuda para consultar tablas que contienen información adicional sobre los tickets y nos permitirán completar algunos campos que no vengan informados desde el origen y sean necesarios, como por ejemplo, campos de provincia o tecnología.

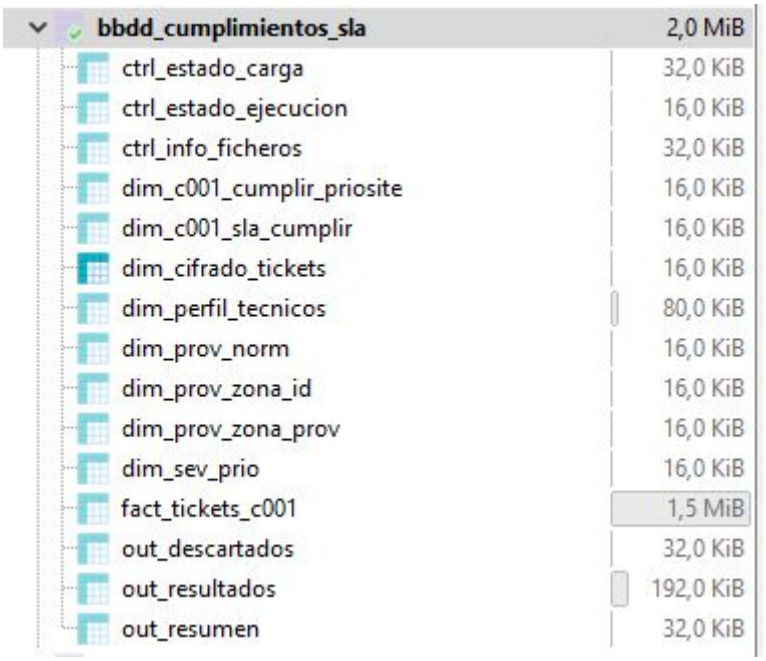


Figura 5.3: Tablas BBDD Interna

Base de Datos Interna

La BBDD principal a partir de la cual se centra este proyecto es *bbdd_cumplimientos_sla* y contiene 4 tipos de tablas distintas, hechos, dimensiones, resultados y control. Ya hemos descrito el contenido de las tres primeras al ver el tipo de ficheros. Sin embargo, el último tipo de tabla se ha creado para poder realizar un seguimiento del estado de los procesos de carga y cálculo de cumplimientos.

- *ctrl_estado_carga*: Esta tabla contendrá tantas filas como ficheros puedan ingestarse. Cada registro de la tabla permitirá conocer información sobre el estado de la última carga del fichero, como el resultado, detalles del log, fechas de modificación o localización de los ficheros tras la ejecución.
- *ctrl_estado_ejecución*: Esta tabla se utilizará para reailizar un seguimiento de las ejecuciones realizadas del proceso ETL (cálculo de cumplimientos). Cada registro de la tabla corresponderá a una ejecución distinta y contendrá información sobre la fecha de ejecución, tipo de contrato o sla (en caso de existir más de uno en un futuro), tipo de ejecución (por defecto la ejecución será mensual, pero podría variar), periodo del cálculo, resultados, etc.
- *ctrl_info_ficheros*: No se utiliza para controlar el estado de los procedimientos, sino para almacenar metadatos de los ficheros que se ingestarán a través de NIFI. Cada entrada de la tabla corresponderá a uno de estos ficheros.

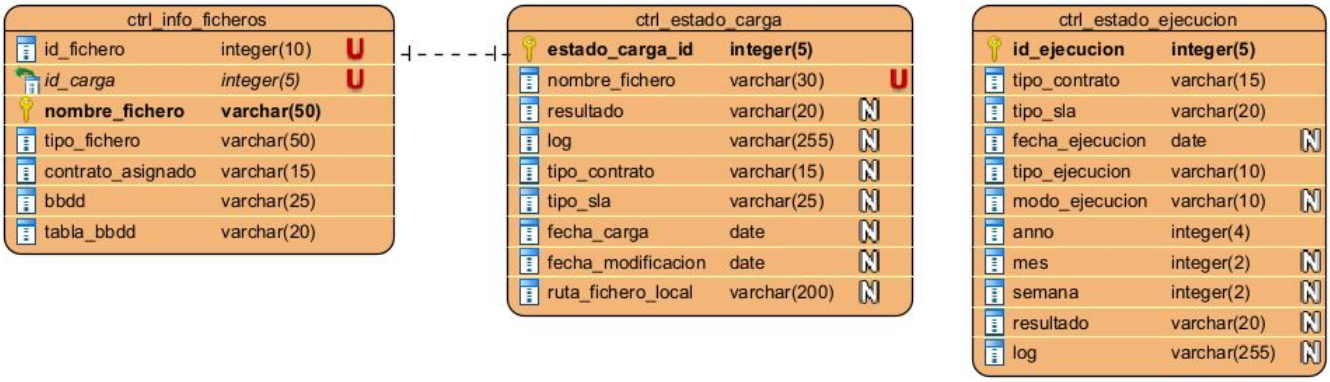


Figura 5.4: Tablas de control

Bases de Datos Externas

- BBDD Inventario: Se trata de una BBDD que sirve para inventariar elementos de red en los cuales se producen las incidencias. Esta contiene varias tablas con información sobre estos elementos, como: su nombre, datos sobre su localización (Altitud, Latitud, Provincia, País, etc.) estado, modelo, tipo y categoría de la tecnología, etc. Las tablas que utilizaremos son: cdr_equipment (con 20 columnas de las que utilizaremos 2), cdr_sites (con 43 columnas de las que utilizaremos 2), familias_tech (utilizados todos los campos) y la tabla dim_emp_fijo (utilizados 2 de 20 columnas). Los campos que contienen identificadores de elementos de red son: 'Name', 'SiteName' y 'elemento_red'.

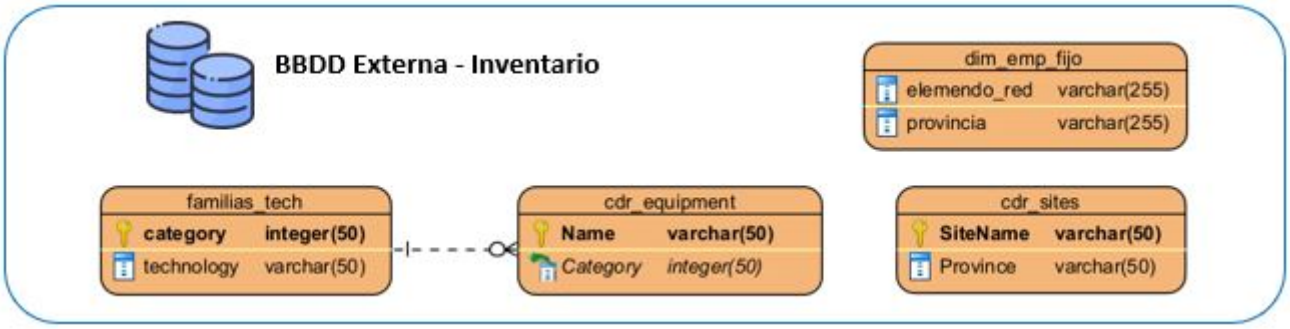


Figura 5.5: Tablas BBDD Inventario

- BBDD Registro Tickets: Esta BBDD contiene tablas que recogen información más detallada sobre las acciones que se han realizado sobre cada ticket, estados por los que ha pasado, usuarios que han intervenido, etc. Utilizaremos todas las columnas de las tablas *dim_estado_ticket* y *tickets_estados*

Figura 5.6: Tablas BBDD Tickets

5.5. Especificación de campos de las tabla.

CTRL_ESTADO_CARGA(estado_carga_id, nombre_fichero, resultado, log, tipo_contrato, tipo_sla, fecha_carga, fecha_modificacion, ruta_fichero_local)

- estado_carga_id: identificador único que se asigna a cada uno de los ficheros que pueden cargarse en la BBDD.
- nombre_fichero: cadena que identifica a cada fichero que puede cargarse en la BBDD.
- resultado: campo en el que se almacenará el resultado del proceso de carga.
- log: mensaje de error que se guardará en caso de que el proceso de carga falle.
- tipo_contrato: identificador del contrato al que pertenece el fichero.
- tipo_sla: tipo de contrato SLA al que pertenece el fichero.
- fecha_carga: almacenará la fecha en la que se ha cargado el fichero en la BBDD,
- fecha_modificacion: fecha en la que se ha iniciado el proceso de carga.
- ruta_fichero_local: especifica la ruta donde se ha almacenado el fichero tras el proceso de carga.

CTRL_ESTADO_EJECUCIÓN(id_ejecucion, tipo_contrato, tipo_sla, fecha_ejecucion, tipo_ejecucion, modo_ejecucion, anno, mes, semana, resultado, log)

- id_ejecucion: identificador único que se asigna a cada ejecución realizada.
- tipo_contrato: identificador del contrato sobre el que se realiza la ejecución.
- tipo_sla: identificador del sla sobre el que se realiza la ejecución.
- fecha_ejecucion: fecha en la que se ha realizado la ejecución.
- tipo_ejecucion: tipo de perioricidad de la ejecución.
- modo_ejecucion: forma en la que se ha comenzado la ejecución.
- anno: año sobre el que se quiere realizar la ejecución.
- mes: mes sobre el que se quiere realizar la ejecución.
- semana: semana sobre la que se quiere realizar la ejecución.
- resultado: campo en el que se almacenará el resultado del proceso.
- log: mensaje de error que se guardará en caso de que el proceso falle.

CTRL_INFO_FICHEROS(id_fichero, id_carga, nombre_fichero, tipo_fichero, contrato_asignado, bbdd, tabla_bbdd)

- id_fichero: identificador único asignado a cada fichero.
- id_carga: identificador único que corresponde a una entrada de la tabla 'ctrl_estado_carga'
- nombre_fichero: cadena que contiene los nombres que deben tener los ficheros a cargar.

- tipo_fichero: tipo de datos que contiene el fichero (dimensión o hechos)
- contrato_asignado: identificador del contrato en caso de tener asignado un único contrato.
- bbdd: nombre de la BBDD a la que pertenece el fichero
- tabla_bbdd: nombre de la tabla donde se cargarán los datos de dicho fichero.

DIM_C001_CUMPLIR_PRIOSITE(priority, site_category, presence_time, restoracion_time, resolucion_time, fecha_modificacion)

- priority: campo que describe la prioridad de un ticket
- site_category: identificador de la zona/site a la que está asignada un ticket.
- presence_time: tiempo de presencia que debe cumplir un ticket según su prioridad y site (CAMPO SIN UTILIDAD).
- restoracion_time: tiempo de restauracion que debe cumplir un ticket reabierto según su prioridad y site.
- resolucion_time: tiempo de resolucion que debe cumplir un ticket según su prioridad y site.
- fecha_modificacion: fecha en la que se ha realizado la última modificación sobre la tabla (última carga).

DIM_C001_SLA_CUMPLIR(sla_id, sla_definition, sla_definition, sla_definition, fecha_modificacion)

- sla_id: identificador del KPI a cumplir
- sla_definition: descripción del KPI
- sla_target_definition: descripción del objetivo a cumplir.
- sla_percentage: porcentaje de cumplimiento del KPI.
- fecha_modificacion: fecha en la que se ha realizado la última modificación sobre la tabla (última carga).

DIM_CIFRADO_TECNICOS (v_num, v_alfa, fecha_modificacion)

- v_num: número decimal asociado a cada una de las letras del abecedario.
- v_alfa: letra del alfabeto español por la que se sustituirá cada dígito.
- fecha_modificacion: fecha en la que se ha realizado la última modificación sobre la tabla (última carga).

DIM_PERFIL_TECNICOS (id_usuario_anon, id_usuario, perfil_usuario, fecha_modificacion)

- id_usuario_anon: identificador único compuesto por caracteres alfanuméricos asociado a cada técnico.
- id_usuario: identificador único que se le asigna a cada técnico a partir de sus datos personales.
- perfil_usuario:
- fecha_modificacion: fecha en la que se ha realizado la última modificación sobre la tabla (última carga).

DIM_PROV_NORM(provincia, provincia_def, fecha_modificacion)

- provincia: nombre de provincia sin normalizar.
- provincia_def: nombre de provincia normalizado.
- fecha_modificacion: fecha en la que se ha realizado la última modificación sobre la tabla (última carga).

DIM_PROV_ZONA_ID(zona_id, nombre_zona, fecha_modificacion)

- zona_id: identificador de las zonas a las que pueden estar asignadas las provincias.
- nombre_zona: campo que describe la zona.
- fecha_modificacion: fecha en la que se ha realizado la última modificación sobre la tabla (última carga).

DIM_PROV_ZONA_ID(provincia_def, zona_id, fecha_modificacion)

- provincia_def: nombre de provincia normalizado.
- zona_id: identificador de la zona a la que pertenece la provincia.
- fecha_modificacion: fecha en la que se ha realizado la última modificación sobre la tabla (última carga).

DIM_SEV_PRIO(prioridad, severidad, tecnologia, fecha_modificacion)

- prioridad: campo que describe la prioridad de un ticket a partir de su severidad y tecnología.
- severidad: campo que describe la prioridad de un ticket.
- tecnologia: campo que describe la tecnología de un ticket.
- fecha_modificacion: fecha en la que se ha realizado la última modificación sobre la tabla (última carga).

FACT_TICKETS_C001(NumerodelTicket, FechaDeAperturaDelTicket, FechaFinPresencia, IDRemedy, ID_IPM, SeveridadTicket, Resol1, Resol2, Resol3, TipodeOperacion, Datosdelticket_Causa, ElementodeRed, EstadoDelTicket, Tecnologia, TipoTicket, Provincia.

- NumerodelTicket: identificador único que se asigna a cada incidencia que entra al sistema denominada ticket.
- FechaDeAperturaDelTicket: Fecha en la que el ticket entro al sistema por primera vez
- FechaFinPresencia: Fecha en la que
- IDRemedy: identificador único del elemento de red (Sin USO)
- ID_IPM: identificador único del elemento de red (Sin USO)
- SeveridadTicket: valor numérico que indica el grado de severidad de la incidencia.

- Resol1: Campo descriptivo que contendrá información sobre la forma en la que se ha solucionado el ticket.
- Resol2: Campo descriptivo que contendrá información sobre la forma en la que se ha solucionado el ticket.
- Resol3: Campo descriptivo que contendrá información sobre la forma en la que se ha solucionado el ticket.
- TipodeOperacion: tipo de acción que se debe realizar sobre la incidencia.
- Datosdelticket_Causa: Campo Adicional donde se añadirán comentarios y descripciones.
- ElementodeRed: Identificador único del elemento que ha sufrido la incidencia y que ha originado el ticket.
- EstadoDelTicket: estado en el que se encuentra el ticket/incidencia.
- Tecnologia: campo sin normalizar que describe la tecnología de un ticket.
- TipoTicket: identificador del tipo o categoría del ticket.
- Provincia: provincia sin normalizar donde se encuentra el elemento asociado a la incidencia/ticket.

OUT_DESCARTADOS(id_ticket_cod, motivo_descarte, tipo_contrato, tipo_sla, sem_sol, mes_sol, anno_sol, fecha_carga)

- id_ticket_cod: identificador único del ticket anonimizado.
- motivo_descarte: descripcion breve del motivo por el cual se ha descartado el ticket.
- tipo_contrato: identificador del contrato al que está asociado el ticket.
- tipo_sla: identificador del sla del que se ha ejecutado el cálculo.
- sem_sol: semana en la que se solucionó el ticket
- mes_sol: mes en el que se solucionó el ticket.
- anno_sol: año en el que se solucionó el ticket.
- fecha_carga: fecha en la que se ha cargado el ticket en la tabla.

OUT_DESCARTADOS(id_ticket_cod, fecha_apertura, fecha_solucionado, fecha_ult_reapertura, fecha_ult_solucionac id_usuario_anon, tiempo_solucion, tiempo_sol_reap, estado, prioridad, categoria_site, zona_site, provincia, pais, reaperturas, obj_sla_resol, cumple_SLA_resol, obj_sla_rest, cumple_SLA_rest, cumple_SLA_concat, sla_id_resol, sla_id_contac, sla_id_rest, tipo_sla, sem_sol, mes_sol, anno_sol, tipo_ejecucion, fecha_carga)

- id_ticket_cod: identificador único del ticket anonimizado.
- fecha_apertura: fecha en la que se abrió la incidencia o ticket.
- fecha_solucionado: fecha en la que se solucionó el ticket por primera vez.
- fecha_ult_reapertura: fecha de la que se abrió el ticket por última vez.
- fecha_ult_solucionado: fecha en la que se solucionó el ticket por última vez.

- id_usuario_anon: identificador alfanumérico único asociado al técnico encargado de solucionar la incidencia.
- tiempo_solucion: tiempo total en el que se solucionó el ticket por primera vez.
- tiempo_sol_reap: tiempo total en el que se solucionó el ticket por última vez
- estado: estado del ticket solucionado.
- prioridad: identificador numérico que describe la prioridad del ticket.
- categoria_site: identificador numérico que describe la zona del ticket.
- zona_site: nombre de la zona asociada al ticket.
- provincia: provincia donde se ha dado la incidencia.
- pais: pais (para fines visuales)
- reaperturas: indicador para saber si el ticket tiene o no reaperturas. Los valores posibles serán 'SI' o 'NO'
- obj_sla_resol: tiempo máximo que debe tardar el técnico en solucionar la incidencia por primera vez. Se trata del objetivo de tiempo de resolución.
- cumple_SLA_resol: indicador para saber si se ha cumplido o no el objetivo anterior. Los valores posibles serán 'SI' o 'NO'
- obj_sla_rest: tiempo máximo que debe tardar el titécnico en solucionar la última reapertura del ticket. Se trata del objetivo de tiempo de restauración.
- cumple_SLA_rest: indicador para saber si se ha cumplido o no el objetivo anterior. Los valores posibles serán 'SI' o 'NO'
- cumple_SLA_concat: indicador para saber si el ticket se ha resuelto en un primer contacto. Los valores posibles serán 'SI' o 'NO'
- sla_id_resol: identificador del KPI asociado al tiempo de resolución.
- sla_id_contac: campo que contiene, si procede, el identificador del KPI03.
- sla_id_rest: campo que contiene, si procede, el identificador del KPI04.
- sem_sol: semana en la que se solucionó el ticket
- mes_sol: mes en el que se solucionó el ticket.
- anno_sol: año en el que se solucionó el ticket.
- dia_sem: caracter que indica el día de la semana en el que se solucionó el ticket.
- fecha_carga: fecha en la que se ha cargado el ticket en la tabla.

OUT_RESUMEN(mes_sol, anno_sol, sem_sol, total_tickets, tickets_calculo, tickets_descarte, %tickets_con_reap, num_KPI01, %KPI01, cumple_KPI01, num_KPI02, %KPI02, cumple_KPI02, num_KPI03, %KPI03, cumple_KPI03, num_KPI04, %KPI04, cumple_KPI04, obj_KPI01, obj_KPI02, obj_KPI03, obj_KPI04)

- mes_sol: mes sobre el que se ha realizado la ejecución.
- anno_sol: año sobre el que se ha realizado la ejecución.
- sem_sol: semana (si fuera el caso) en la que se ha realizado la ejecución.
- total_tickets: cantidad total de tickets que se han resuelto el en periodo de la ejecución, incluyendo tickets válidos y descartados.
- tickets_calculo: cantidad de tickets válidos que se han utilizado para realizar los cálculos.
- tickets_descarte: cantidad de tickets que se han descartado.
- %tickets_con_reap: Porcentaje de tickets válidos que han tenido reaperturas
- num_KPI01: cantidad de tickets asociados al KPI01
- %KPI01: porcentaje tickets que han cumplido el KPI01 en el periodo de la ejecución.
- cumple_KPI01: indicador para saber se ha cumplido el KPI01 en el periodo de la ejecución.
- num_KPI02: cantidad de tickets asociados al KPI01
- %KPI02: porcentaje tickets que han cumplido el KPI02 en el periodo de la ejecución.
- cumple_KPI02: indicador para saber se ha cumplido el KPI02 en el periodo de la ejecución.
- num_KPI03: cantidad de tickets asociados al KPI03
- %KPI03: porcentaje tickets que han cumplido el KPI03 en el periodo de la ejecución.
- cumple_KPI03: indicador para saber se ha cumplido el KPI03 en el periodo de la ejecución.
- num_KPI04: cantidad de tickets asociados al KPI04
- %KPI04: porcentaje tickets que han cumplido el KPI04 en el periodo de la ejecución.
- cumple_KPI04: indicador para saber se ha cumplido el KPI04 en el periodo de la ejecución.
- obj_KPI01: porcentaje de tickets que deben a cumplir para el KPI01.
- obj_KPI02: porcentaje de tickets que deben a cumplir para el KPI02.
- obj_KPI03: porcentaje de tickets que deben a cumplir para el KPI03.
- obj_KPI04: porcentaje de tickets que deben a cumplir para el KPI04.

5.6. Estructura de Directorios

Se ha diseñado una estructura de directorios para alojar los procesos de cálculo y ficheros utilizados. Esta estructura tendrá un directorio raíz situado en la ruta: C:\My_TFG\src\app \tools\ del cual colgarán en todo momento al menos dos directorios, uno denominado 'comunes', que contendrá ficheros compartidos (dimensiones) por al menos 2 contratos, y por otro tendremos directorios específicos para cada contrato, los cuales tendrán la siguiente nomenclatura: 'slas_CXXX', donde las XXX corresponderán a un contrato específico.

Directorio raiz:

- \comunes
  - \bin
  - \dat
    - \mailbox
    - \notloaded
    - \processed
- \slas_C001
  - \bin
  - \dat
    - \logs
    - \output
    - \tmp
  - \env_C001

Descripción de directorios

- **comunes**: Directorio raiz que alberga ficheros y procesos comunes a dos o más contratos a la vez.
- **slas_CXXX**: Directorio raiz donde se almacenan los ficheros de cálculo específicos para un contrato CXXX, donde las XXX corresponden al identificador de cada contrato.
- ***\bin**: Directorios donde se almacenan el código script y los procesos Python.
- ***\dat**: Directorio donde se almacenan los ficheros de entrada y salida.
- ***\env_c001**: Directorio donde se almacenan el entorno virtual de python.
- ***\dat\mailbox**: Directorio en el cuál se depositan temporalmente los ficheros de entrada que serán leídos mediante el proceso de ingesta.
- ***\dat\notloaded**: Directorio destino en el cuál se depositarán los ficheros leídos por NIFI y que no se habrán podido cargar en las tablas de la BBDD por algún fallo.
- ***\dat\processed**: Directorio destino en el que se depositarán los ficheros leídos por NIFI que sí se hayan podido cargar en la BBDD.
- ***\dat\logs**: Directorio que contiene los ficheros de logs donde se pueden ver mensajes de error e información sobre la ejecución.
- ***\dat\output**: Directorio que contiene los ficheros comprimidos con los resultados generados en el proceso ETL, que se utilizarán posteriormente para la visualización.
- ***\dat\tmp**: Directorio temporal donde se almacenan los anteriores resultados, antes de comprimirlos y moverlos a la carpeta 'output'. Este directorio se purgará tras cada ejecución.

5.7. Diseño Ingesta en NIFI

Como se ha visto en el Apartado 3, utilizaremos la Herramienta NIFI para realizar la ingesta inicial de datos. Antes de explicar el diseño planteado, se van a describir algunos términos para comprender mejor el funcionamiento de esta herramienta. Los 4 conceptos básicos que deben conocerse son: *FlowFile, Procesador, Conector y Controlador*, [19].

En primer lugar, el *Flowfile* es el paquete de información que se mueve a través del flujo de datos implementado por Nifi. Este paquete representa cada objeto en movimiento y está dividido en dos partes, atributos y contenidos.

- Los **Atributos**: son pares clave-valor que representan los metadatos asociados a los datos. Los más comunes son: el nombre del archivo (filename), su ruta o un identificador único (uuid). Pueden agregarse mas atributos.
- El **Contenido**: contiene un puntero que hace referencia a los datos almacenados localmente. Estos componentes donde se almacenan los datos son los llamados repositorios de contenido (Content Repository). Esto mejora el rendimiento del sistema ya que reduce los accesos a disco. Muchos procesadores no modifican el contenido de los flowfiles, por lo que no necesitan cargar su contenido en memoria.

Figura 5.7: Estructura Flowfile

Procesadores. Son los componentes principales de nifi, encargados de realizar las operaciones principales sobre los datos, como enrutamientos, extracciones, transformaciones o cargas. Sus principales características son:

- Proporcionan la interfaz para acceder a los atributos de los flowfiles y a su contenido.
- Existen más de 280 procesadores preinstalados, aunque pueden implementarse nuevos procesadores personalizados.
- Es posible programar su ejecución mediante eventos, tiempos predefinidos o mediante cron.
- Tienen distintas conexiones de salida, por ejemplo success(éxito)/failure(fallo), retry (reintento), matched/unmatched, etc...

Conectores/Conexiones: son los componentes que enlazan los procesadores entre sí, se comportan como colas y permiten que estos interactúen a distintas velocidades. Son altamente configurables, pueden priorizar los Flowfiles en cola y establecer límites de cargas de trabajo.

Finalmente, el *Controlador (Flowfile Controller)*, quien ejecuta todo el flujo de datos, actúa como intermediario y se encarga de asignar y administrar los recursos entre los componentes anteriores. Al igual que ocurre con los procesadores, existen gran cantidad de servicios de controlador predefinidos.

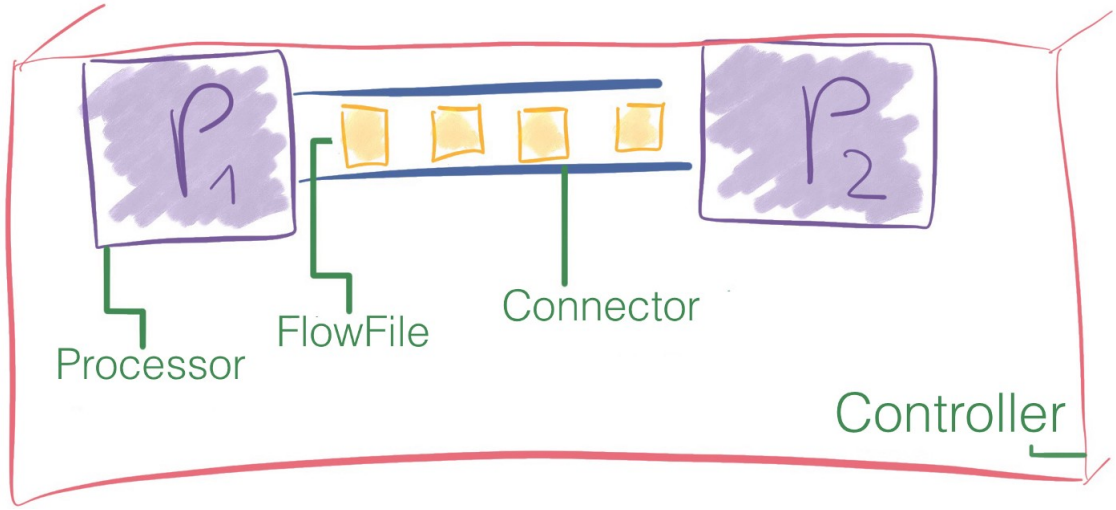


Figura 5.8: Componentes esenciales NIFI

Detalles de la Ejecución

El flujo de Nifi se ejecutará de forma continua a la espera de recibir un evento que desencadene la ingesta. Este evento será recibir por SFTP un archivo csv en un directorio local (*\dat\mailbox). Existen dos clases distintas de archivos que se pueden recibir, como hemos visto anteriormente:

- Ficheros de Dimensión: no existe ninguna periodicidad para este tipo de archivos, por lo tanto se actualizarán en la BBDD cuando se realice alguna modificación sobre ellos.
- Ficheros de Hechos: Estos ficheros son los que nutren todo el proceso de cálculo de slas para cada contrato. Son ficheros que se reciben con cierta periodicidad y que contienen toda la información sobre los tickets generados durante dicho periodo. En este caso la periodicidad será mensual y se tratará de un único fichero que se recibirá en los primeros días de cada mes.

Tras recibir estos, se realizará una consulta SQL sobre la tabla *ctrl_info_ficheros* para conocer los metadatos asociados al fichero leído, como su identificador de carga, el tipo de fichero, contrato asignado, bbdd y tabla de a la que deben ingestarse. El identificador de carga será necesario para actualizar la tabla *ctrl_estado_carga*, creada para realizar un seguimiento del proceso de ingesta. El resto de metadatos nos servirán para realizar validaciones y enrutamientos a través de los procesadores de NIFI para poder cargar los datos en las tablas correspondientes de cada BBDD. La carga de los datos será total, se borrarán por completo los datos antiguos y se insertarán los nuevos. Además, los ficheros se volverán a almacenar localmente, en los directorios *\dat\processed o *\dat\notloaded, dependiendo de si la ingesta ha tenido exito o no, respectivamente.

5.8. Definición de métricas y SLAs a cumplir

Antes de comenzar a explicar el diseño del proceso de cálculo, se van a explicar algunas de las métricas que se han decidido calcular y los KPIs asociados al SLA que se deben cumplir.

5.8.1. Métricas

En primer lugar se han definido las métricas que queremos obtener para poder realizar un análisis de los datos y crear informes y cuadros de mando con Power BI. Estas métricas servirán para medir el cumplimiento de slas, y son:

Tiempos de Resolución

La primera y principal métrica que queremos obtener para este proyecto será el tiempo de resolución, es decir, el tiempo que tarda el proveedor en resolver una incidencia (ticket) desde que esta entra al sistema hasta que se resuelve o da por cerrado. En ocasiones puede ocurrir que un ticket no se haya solucionado correctamente tras una primera interacción y sea necesario reabrirlo. Por ello, obtendremos dos tipos de tiempos de resolución distintos:

1. Tiempo primera solución: Para calcular este tiempo necesitaremos conocer la fecha en la que el ticket se marco como 'Solucionado' o 'Cerrado' por primera vez.

2. Tiempo última resolución: para este tiempo obtendremos la fecha en la que el ticket se solucionó por última vez. Es posible que un ticket tenga más de una reapertura, por lo que utilizaremos únicamente la última.

Para poder distinguir ambos más fácilemente, a partir de ahora nos referiremos a ellos como tiempo de solución y tiempo de restauración respectivamente.

Nº de Reaperturas

Para cada ticket se van a calcular el número total de reaperturas. La frecuencia con la que vuelven a abrir los tickets es un posible indicador de que los agentes no están resolviendo por completo los problemas de los clientes. Esto puede deberse a que los agentes necesiten mayor entrenamiento, cuanto mayor sea su entrenamiento mayor será su productividad, o bien puede significar que no hay suficientes agentes y deban contratarse más. Para realizar un seguimiento sobre esta métrica calcularemos el % de tickets con reapertura.

Volumen de tickets solucionados por día de la semana

Para cada ticket, se va a calcular el día de la semana que se solucionó definitivamente. Esta métrica ayuda a obtener estadísticas que muestran cuando un servicio está más ocupado y cuando se necesitan más recursos o estar más disponibles.

Volumen de tickets solucionados por agente/técnico

Para cada ticket deberemos almacenar el agente encargado de solucionarlo. Esto permitirá conocer el volumen de trabajo que tiene cada agente. Además, al conocer el agente se podrá realizar una media de tiempos de resolución por técnico para conocer aquellos que están resolviendo las incidencias más rápidamente.

5.8.2. SLAs a cumplir y KPIs

Las anteriores métricas nos ayudan a visualizar el rendimiento general que tiene la empresa en relación a la resolución de incidencias. Sin embargo el objetivo principal de este proyecto es comprobar el grado de cumplimiento de los slas definidos. Para poder obtener esta información primero debemos conocer cual es el sla asociado a cada ticket, estos serán tiempos de resolución que deben cumplir.

Cada uno de los tickets del contrato c001 tiene asociado un sla dependiendo de su nivel de prioridad y el site en el que se encuentre. Existen 5 niveles de prioridad distintos y 4 sites (Centro, Norte, Este, Oeste) diferentes a los que puede estar asignado cada uno. En la figura 5.9 se puede ver como están distribuidas las provincias por site y en la figura 5.10 se pueden ver los slas asociados a cada prioridad/site.

Figura 5.9: Zonas Geográficas - Sites

dim_c001_cumplir_priosite (20r × 5c)

priority	site_category	restoration_time	resolution_time	fecha_modificacion
Prio1	1	96	168	2022-06-08 23:11:54
Prio1	2	96	168	2022-06-08 23:11:54
Prio1	3	216	336	2022-06-08 23:11:54
Prio1	4	216	336	2022-06-08 23:11:54
Prio2	1	96	168	2022-06-08 23:11:54
Prio2	2	96	168	2022-06-08 23:11:54
Prio2	3	216	336	2022-06-08 23:11:54
Prio2	4	216	336	2022-06-08 23:11:54
Prio3	1	216	336	2022-06-08 23:11:54
Prio3	2	216	336	2022-06-08 23:11:54
Prio3	3	360	504	2022-06-08 23:11:54
Prio3	4	360	504	2022-06-08 23:11:54
Prio4	1	360	504	2022-06-08 23:11:54
Prio4	2	360	504	2022-06-08 23:11:54
Prio4	3	480	720	2022-06-08 23:11:54
Prio4	4	480	720	2022-06-08 23:11:54
Prio5	1	480	720	2022-06-08 23:11:54
Prio5	2	480	720	2022-06-08 23:11:54
Prio5	3	550	800	2022-06-08 23:11:54
Prio5	4	550	800	2022-06-08 23:11:54

Figura 5.10: Contenido tabla *dim_c001_cumplir_priosite*

Cumplimiento SLAs

El objetivo principal de este proceso es conocer si se están incumpliendo unos SLAs. Por ello, para cada ticket, debemos conocer a que SLA corresponden y si se están o no cumpliendo las condiciones establecidas. Para poder realizar un seguimiento sobre esta métrica podremos observar el % de cumplimiento.

Para el contrato c001 se han establecido cuatro KPIs que deben cumplirse:

- **KPI01**: Los tickets con prioridad 1 y 2 deben cumplir el sla de tiempo de solución en un 99 %.
- **KPI02**: Los tickets con prioridad 3, 4 y 5 deben cumplir el sla de tiempo de solución en un 97 %.
- **KPI03**: El 95 % de los tickets deben haberse solucionado en una primera interacción, o lo que es lo mismo, el número de tickets con reaperturas no debe superar el 5 %.
- **KPI04**: De los tickets que hayan tenido apertura, el 90 % debe haber cumplido el sla de restauración (tiempo última resolución).

En la siguiente tabla se describen estos KPIs.

dim_c001_sla_cumplir (4r × 5c)

sla_id	sla_definition	sla_target_description	sla_percentage	fecha_modificacion
KPI01	First Time Resolution Prio1-Prio2	? 1,00% cases out of SLA	1	2022-06-08 23:10:17
KPI02	First Time Resolution Prio3-Prio4-Prio5	? 3,00% cases out of SLA	3	2022-06-08 23:10:17
KPI03	First Contact Resolution	? 5,00% cases out of SLA	5	2022-06-08 23:10:17
KPI04	Last Time Resolution (Restoration)	? 10,0% cases out of SLA	10	2021-12-17 13:41:37

Figura 5.11: Contenido tabla *dim_c001_sla_cumplir*

5.9. Diseño de la ETL - cálculo de cumplimientos

En este apartado se van a describir las acciones llevadas a cabo para realizar el proceso de automatización de los SLAs y KPIs definidos.

Gestión del Código

El procecimiento ETL se ha diseñado para permitir dos tipos de ejecuciones, automática y manual. La primera será programada y ejecutada a través de *cron*, mientras que la manual se podrá lanzar en cualquier momento. En ambos casos, la secuencia de ejecución será similar y estará compuesta por distintos ficheros de tipo script (*.sh) y Python (*.py), que estarán almacenados en el directorio `C:\My_TFG\src\app\tools\slas_C001\bin\*`

1. **calculo_slas_c001.sh**: El primer proceso sh se encargará de activar el entorno de trabajo necesario para la ejecución y realizará las llamadas al resto de procesos, que serán los encargados de generar los resultados.
2. **inicio_ejecucion.py**: Comprobará que sea posible iniciar la ejecución del cálculo, es decir, que existan datos cargados para el periodo deseado. Devolverá el control al script sh con el resultado de esta comprobación para que pueda continuar con la ejecución o finalizarla en caso de error.
3. **calculo_slas_c001.py**: Por último tenemos el fichero encargado del grueso de la ejecución. Accederá a las BBDDs para recoger los datos de las tablas, realizará transformaciones y purgados sobre estos, ejecutará el cálculo de cumplimientos y almacenará los resultados de la ejecución, tanto en la BBDD como en ficheros. Este devolverá el control al script sh, que desactivará el entorno y finalizará la ejecución. Nos referiremos a este como proceso de cálculo.

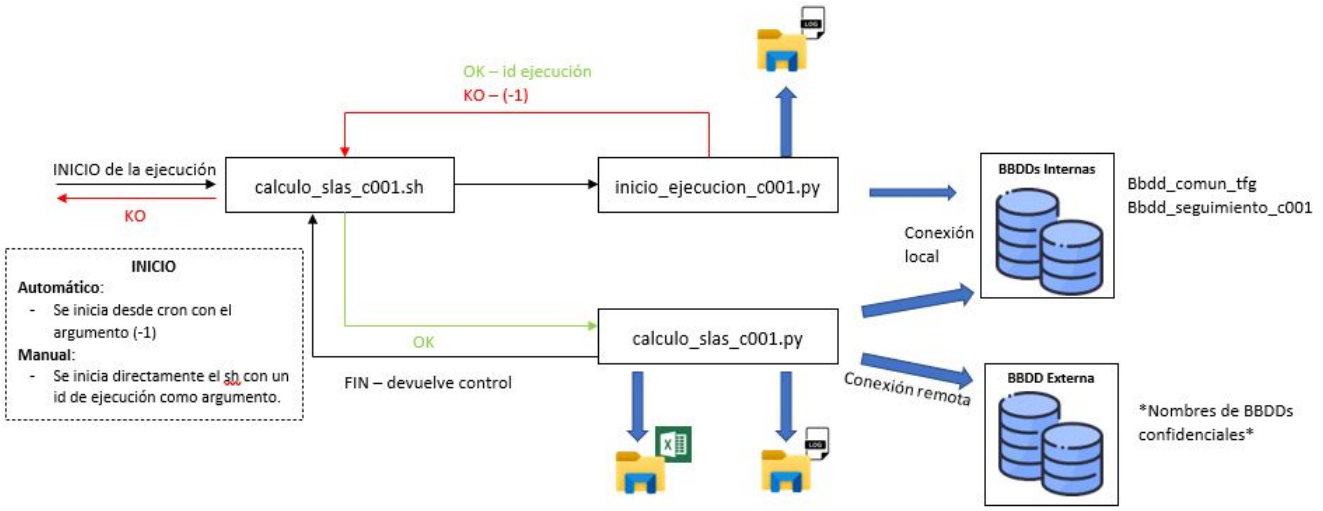


Figura 5.12: Secuencia de Ejecución (ETL)

Los procesos py a su vez harán uso de otros ficheros creados para almacenar constantes y funciones. Estos estarán almacenados en el directorio común, para que en un futuro puedan ser utilizados por más de un contrato. Estos son los ficheros **constantes.py** y **funciones_comunes.py** y estarán almacenados en `C:\My_TFG\src\app\tools\comunes\bin\*`

Explicación de cálculos a realizar

Antes de comenzar a diseñar e implementar el proceso que hemos denominado cálculo, es necesario comprender el alcance de los datos de los que disponemos y conocer que tipo de información adicional es necesaria para que podamos obtener las métricas definidas.

Como hemos visto en la figura 5.10 cada ticket debe cumplir un plazo de resolución de incidencias distinto dependiendo de su nivel de prioridad y el site en el que se encuentre. Estos campos no se encuentran en los datos de origen (tabla de hechos) y deberemos calcularlos a partir de cruces con otras tablas. Cada nivel de prioridad dependerá a su vez de la severidad (campo 'SeveridadTicket' de la tabla de hechos) y tecnología (campo 'Tecnología' de la tabla de hechos) de cada ticket, mientras que el site dependerá de la provincia en la que se encuentre. Sin embargo, alguno de estos campos no podrán utilizarse directamente.

En primer lugar, para obtener la *Tecnología* válida, deberemos utilizar la información que proporcionan las tablas de la BBDD inventario. Los valores que aparecen en el campo de origen son 'FIVF, CABLE' y 'AMBAS', sin embargo, debemos traducir estos a 'Fijo' o 'Movil'. Los tickets con tecnología 'CABLE' se traducen directamente como 'Fijo'. Los tickets 'FIVF' y 'AMBAS' dependerán de la información de la BBDD inventario.

- Debemos hacer un cruce entre las tablas 'cdr_equipment' y 'familias_tech' sobre el campo de categoría, lo que generará una nueva tabla con los campos 'Name', 'Site ID'(será de utilidad para calcular provincia) y 'technology'. El campo 'Name' contiene un identificador de elemento que nos servirá para cruzar con el campo 'ElementodeRed' de la tabla de hechos.
- Si el campo 'technology' es 'TX' se traducirá como 'Movil'
- Si el campo 'technology' es 'Centro Core' se traducirá como 'Fijo'

- Si el campo 'technology' es 'N.A.' y el valor de origen es 'FIVF', el ticket se descartará por no tener tecnología asignada.
- Si el campo 'technology' es 'N.A.' y el valor de origen es 'AMBAS', dejaremos el valor por defecto ('AMBAS').
- Si finalmente algúna tecnología queda sin informar, descartaremos esos tickets, igual que haremos con aquellos que no tengan severidad o cuyo campo 'ElementodeRed' esté vacío o tenga el valor 'Genérico'.

Una vez obtenida la tecnología, podremos cruzar contra la tabla dim_sev_prio. Aquellas que sean 'Fijo' y 'Movil' obtendrán directamente el valor de prioridad, pero para aquellas que sean 'AMBAS' se les asignará la prioridad más alta entre las dos opciones.

El siguiente paso será obtener el site a partir del campo provincia. Este campo existe en el origen pero en muchas ocasiones aparecerá vacío o desnormalizado (por ejemplo, Lleida/Lérida/Lerida).

- Podremos utilizar la tabla cdr_sites de la BBDD inventario para cruzar con el campo 'Site ID' y obtener el campo provincia.
- Para los tickets con tecnología 'Fijo' se podrá cruzar contra la tabla dim_emp_fijo, a través de los campos de elemento de red.
- Si llegados a este punto aun quedaran provincias vacías, se podrá comprobar el campo 'Datosdelticket_Causa' de la tabla de hechos ya que en ocasiones contienen descripciones sobre la localizaciones.
- Los tickets cuya provincia siga vacía se descartarán ya que no podrá obtenerse el site a partir de ellos.
- Para obtener el site cruzaremos la provincia con las dimensiones dim_prov_norm (Para obtener la provincia normalizada) y dim_prov_zona_prov (Para obtener el id del site a partir de la provincia normalizada).

Una vez que tenemos la prioridad y el id del site, podremos obtener los tiempos de solución a cumplir de cada incidencia según sus características. Como vimos en el apartado 5.8.1 de Métricas, calcularemos dos tiempos distintos: Tiempo de primera solución (tiempo de resolucion) y tiempo de última solución (tiempo de restauracion), para ello necesitaremos utilizar la BBDD externa 'tickets' donde se registra cada uno de los estados por lo que han pasado, desde su apertura hasta su cierre definitivo.

- El tiempo de primera solución será la diferencia entre la fecha en la que el ticket se solucionó o cerró por primera vez y la fecha inicial de apertura del ticket. Los KPIs 'KPI01' y 'KPI02' se calcularán sobre este tiempo
- El tiempo de última solución será la diferencia entre la fecha de última reapertura y la fecha de la siguiente vez que se solucionó la incidencia. Es posible que el ticket se haya reabierto en varias ocasiones, así que se buscarán la última vez que ocurrió. El KPI 'KPI04' se calculará sobre este tiempo.

Finalmente, comprobaremos si se han cumplido o no los objetivos de tiempo de resolución(1ª Sol) y restauración (Última Sol). Añadiremos dos nuevas columnas para almacenar el resultado de estas comprobaciones. Si los tiempos superan los objetivos se almacenará 'NO' y en caso contrario 'SI'.

5.10. Diseño Visualización de Datos

Antes de comenzar con la visualización, debemos hacernos ciertas preguntas para conocer los objetivos que queremos conseguir con ella, además estas preguntas nos ayudarán a crear una visualización más intuitiva e interesante. En primer lugar debemos tener claro quién es el público objetivo, que historia quiero contarle o qué respuestas puedo ofrecerle. Nuestro objetivo principal es que este usuario final pueda tomar decisiones más precisas sobre su negocio, partiendo de los datos que se han recopilado y presentado para él.

En segundo lugar, deberemos saber elegir los gráficos según la situación, ya que el mismo formato no siempre será útil en cualquier contexto. Además será importante utilizar patrones previsibles, ya que a los seres humanos nos atraen los indicadores que nos ofrecen información importante a primera vista. Si los patrones son aleatorios, será muy difícil comprender el mensaje que se quiere transmitir y es importante que el usuario pueda pasar de un punto a otro de la visualización de forma rápida y sencilla, sin perder el hilo de lo que está observando.

Otro punto importante es el uso de los colores en la visualización, ya que estos nos permiten transmitir mucha información a simple vista. Pero debemos utilizarlos con cierto equilibrio y criterio, es recomendable no utilizar demasiados colores, ya que el uso indiscriminado de estos provoca que el usuario tenga que realizar un gran esfuerzo para comprender lo que está ocurriendo. El uso de colores debe crear una composición armoniosa para que sea agradable. Sin embargo, en algunas ocasiones es recomendable utilizar colores complementarios para resaltar cierta información o llamar la atención, ya que estos colores se encuentran en puntos opuestos del círculo cromático y causan gran impacto. La mayoría de expertos recomiendan el uso de grises para mostrar datos menos importantes y utilizar colores para indicar al usuario donde debe poner el foco. Otra alternativa sería utilizar colores corporativos, ya que las grandes empresas invierten mucho en su imagen corporativa, pero a veces no son los más adecuados.

Una vez elegidos los colores, es importante mantener una coherencia a lo largo de la visualización para que nuevamente el usuario no tenga que esforzarse en aprender otro esquema de colores distinto.

Para realizar el diseño de nuestra visualización nos centraremos en mostrar al usuario información sobre las métricas de un mes concreto, principalmente será de interés conocer el grado de cumplimiento de los KPIs. Además será interesante conocer ciertos patrones sobre los tickets que no están cumpliendo los slas establecidos, como por ejemplo si estos se están agrupando en zonas concretas o bien si tienen prioridades altas, ya que esto ayudará a aplicar soluciones. Para ello, añadiremos gráficos y filtros que nos ayuden a detectar estos patrones. El diseño realizado deberá ser claro y permitir que en una primera vista el usuario sea capaz de localizar la información relevante. Se crearán varias pestañas para dividir el contenido y dejar estas lo más despejadas posibles.

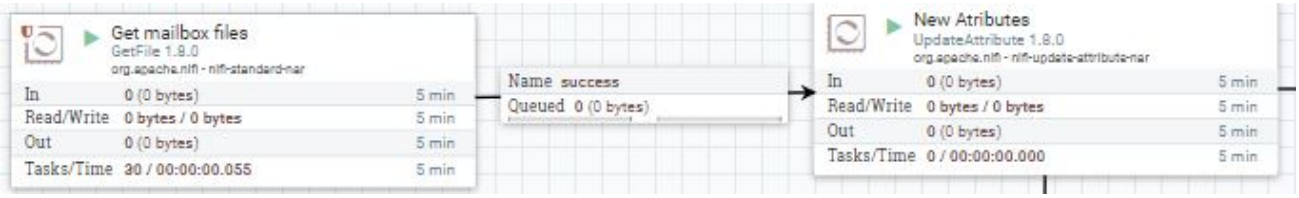
En cuanto a los datos utilizados, se ha elegido uno de los meses que más tickets se hayan manejado, para poder tener mayor variedad en los datos. Tras realizar varias ejecuciones del proceso ETL se ha decidido realizar la visualización sobre el mes de Agosto de 2021, ya que en este mes se resolvieron un total de 447 incidencias de las cuales se descartaron 7 por falta de información.

Capítulo 6

Implementación

En este capítulo se explicarán con mayor detalle cada una de las partes que se han implementado.

6.1. Implementación proceso de Ingesta

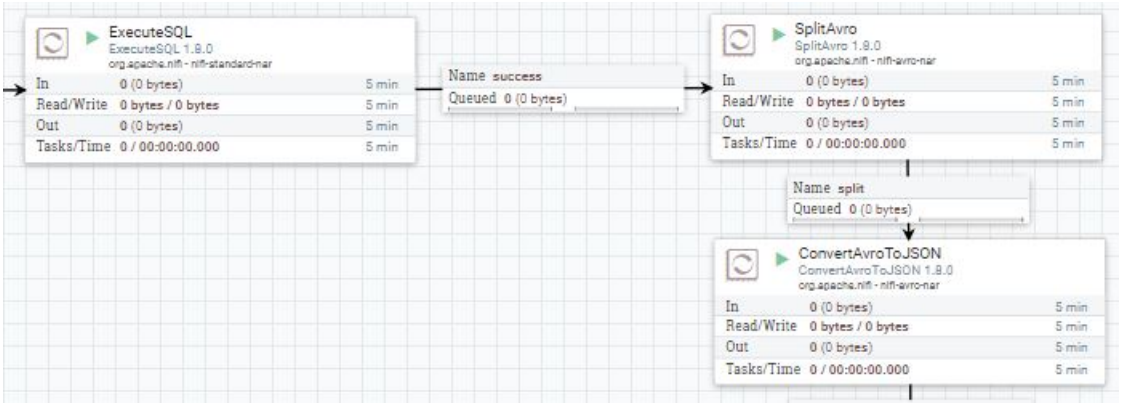


1. El flujo de NIFI comienza cuando se recibe un nuevo fichero en el directorio **data\mailbox*. El primer procesador lo extrae y crea un flujo de datos a partir de él.

2. A continuación extraemos el nombre del archivo (sin la extensión) para poder realizar una consulta sobre la tabla *ctrl_info_ficheros*. Además, en este paso añadiremos otros atributos al flujo de datos, como una marca de tiempo, el directorio de almacenamiento (Se inicializará por defecto con la ruta donde se almacenaría el fichero en caso de error), o la fecha de carga (inicializada a '0000-00-00 00:00:00').

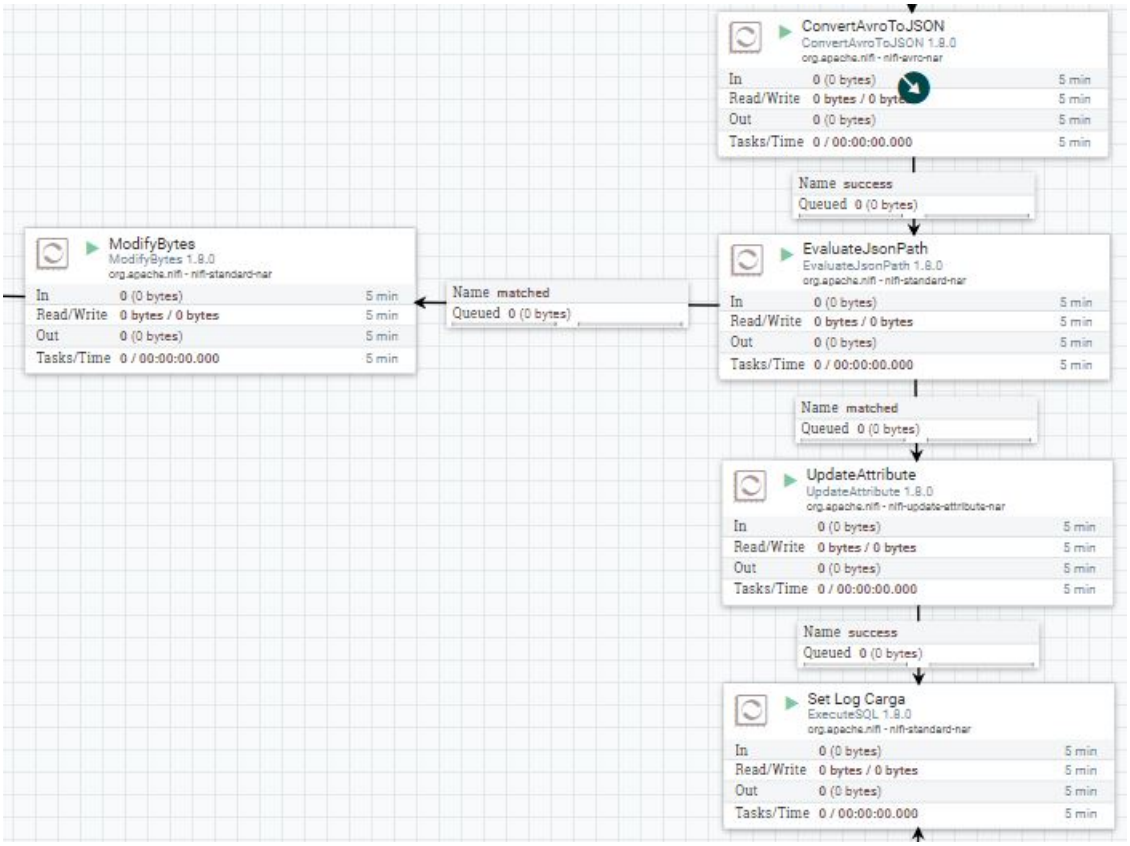
3. Realizamos la consulta SQL sobre la tabla mencionada en el punto anterior, para conocer los metadatos asociados al fichero.

   *SELECT * FROM ctrl_info_ficheros WHERE nombre_fichero='${file_name}';*
   El atributo 'file_name' será el nombre de fichero extraido en el paso anterior.



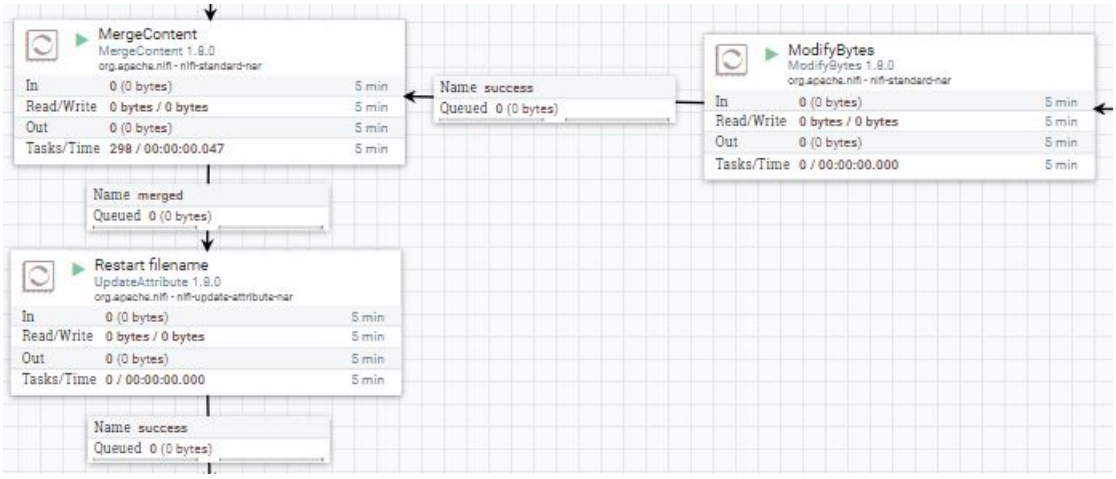
4. Esta consulta generará un contenido en formato AVRO con el resultado de la consulta.

5. Convertimos el contenido a formato JSON y lo evaluamos para almacenar los campos como atributos del flujo de datos. Entre estos atributos se encuentra la bbdd y tabla destino, y el identificador de carga del fichero.



6. A continuación se produce una bifurcación. Por un lado, a partir del identificador de carga que acabamos de leer, actualizamos la entrada de la tabla *ctrl_estado_carga* para indicar que se ha iniciado la ingesta de dicho archivo.

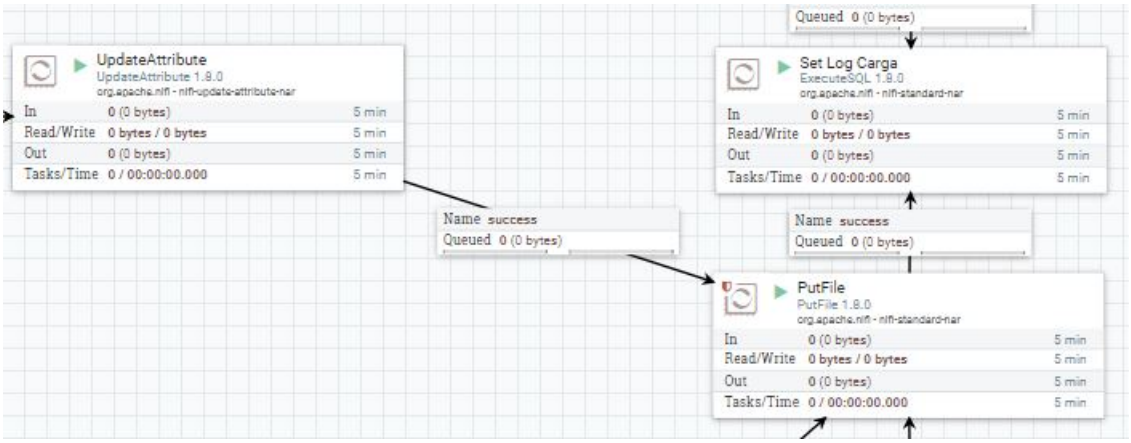
   - Inicializamos el campo 'resultado' a 'WORK IN PROGRESS'
   - Inicializamos el campo fecha_carga con '0000-00-00 00:00:00'.
   - Inicializamos el campo 'ruta_fichero_local con `C:\My_TFG\src\app\tools\comunes\data\notloaded\nomb`



7. Al otro lado de la bifurcación continua el flujo. El siguiente paso será borrar el contenido que se ha generado tras la consulta SQL, puesto que este ya se ha guardado como atributos y no queremos que corrompa los datos que queremos ingestar.

8. A continuación realizamos una unión con el flujo de datos original, que contiene los datos del fichero.

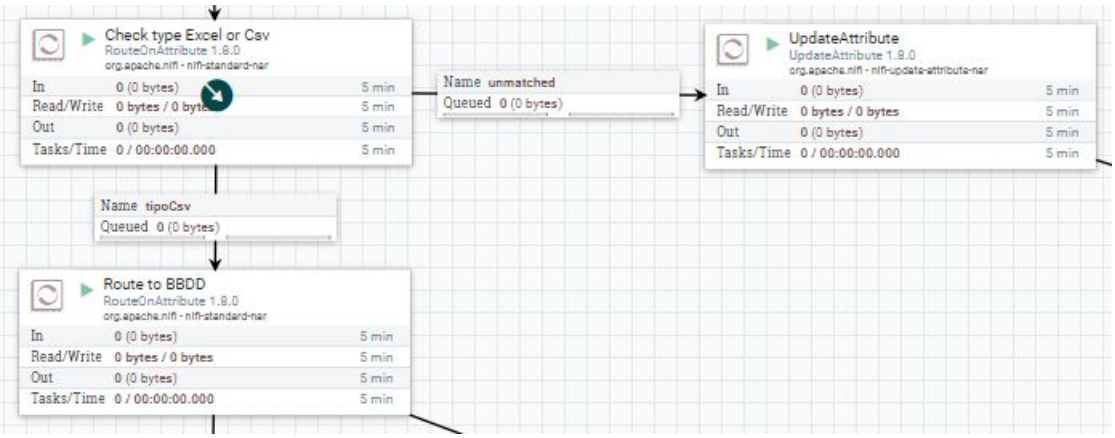
9. Tras la unión, el atributo 'filename' se ha sobreescrito por lo que deberemos restaurar el original. Esto será necesario ya que el procesador encargado de almacenar los ficheros localmente utiliza este atributo para almacenarlos con dicho nombre.



10. Comprobamos que el formato del fichero recibido sea CSV.

11. En caso de que no hayamos recibido el formato deseado, descargamos el archivo en el directorio **data**notloaded* y actualizamos la entrada correspondiende de la tabla *ctrl_estado_carga* indicando que el proceso ha fallado.

    - Actualizamos el campo 'resultado' a 'KO'
    - Insertamos la descripción 'EL FORMATO DE FICHERO NO ES EL ESPERADO (CSV)' en el campo 'log'
    - Mantenemos el campo fecha_carga con el valor '0000-00-00 00:00:00'.
    - Mantenemos el campo 'ruta_fichero_local con `C:\My_TFG\src\app\tools\comunes\data\notloaded\no`

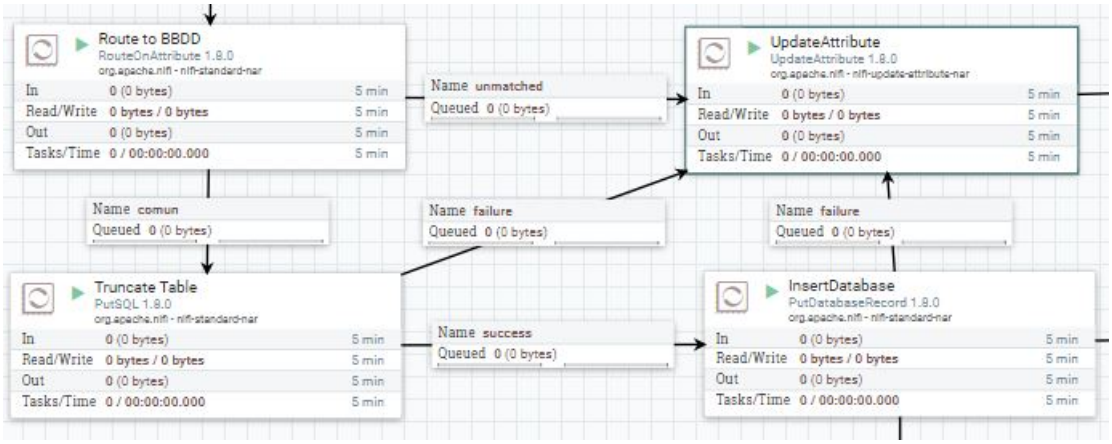


12. Comprobamos que la bbdd donde debemos cargar los datos coincida con el campo de bbdd extraído en la consulta SQL inicial.

13. Realizamos un borrado de datos de la tabla donde deberemos realizar la carga.

14. Insertamos los datos del fichero en la tabla de datos correspondiente.

15. Si alguna de las acciones anteriores relacionadas con la BBDD falla, se descargará el fichero en el directorio **data**notloaded* y se actualizará entrada correspondiende de la tabla *ctrl_estado_carga* indicando que el proceso ha fallado y error ha sido un fallo al operar con la BBDD.



16. Si la carga de datos ha finalizado con éxito, descargamos el fichero en el directorio **data**notloaded* y actualizamos la entrada correspondiende de la tabla *ctrl_estado_carga* indicando que el proceso ha tenido éxito.

    - Actualizamos el campo 'resultado' a 'OK'
    - Actualizamos el campo fecha_carga con la fecha actual de la ejecución
    - Modificamos el campo 'ruta_fichero_local' por `C:\My_TFG\src\app\tools\comunes\data\processed\nombr`

17. Finaliza el proceso de Ingesta.

6.2. Implementación proceso ETL - Cálculo

6.2.1. Proceso 1 - Shell Script

1. Para comenzar, inicializamos las variables que vayamos a necesitar durante la ejecución del script, estas son: rutas de directorios donde se encuentran los procesos que se van a llamar, identicador de la ejecución, marca de tiempo (para añadir al log), datos sobre proceso de cálculo que queremos ejecutar (tipo contrato, tipo sla).

2. Creamos el fichero de log donde iremos añadiendo distintos mensajes sobre toda la ejecución del proceso ETL, tanto errores como información relevante. Le añadimos una cabecera incluyendo la fecha actual y otorgamos todos los permisos al fichero.

3. Activamos el entorno virtual que hemos creado, para que se puedan ejecutar los procesos py con las librerías con las que se han programado.

4. Ejecutamos el primer proceso de python, pasándole como argumento:

    - Argumento 1: Ruta donde se encuentra nuestro fichero log.
    - Argumento 2: Id de la ejecución.
    - Argumento 3: Nombre del contrato al que está asociado el sla.
    - Argumento 3: Nombre del sla que queremos calcular.

5. El proceso anterior comprobará que existan datos cargados en la bbdd para comenzar el cálculo. De ser así, este devolverá de nuevo el id de la ejecución, que guardaremos de nuevo en una variable llamada 'resultado'.

    - En caso de estar realizando una ejecución manual, el id de la ejecución no habrá cambiado.

```
#!/bin/bash

#INICIALIZACION RUTAS FICHEROS
#Las rutas de los ficheros/scripts están definidas desde el subsistema de windows para linux.
rutascomun="/mnt/c/My_TFG/src/app/tools/comunes/bin/"
rutascripts="/mnt/c/My_TFG/src/app/tools/slas_C001/bin/"
rutasdata="/mnt/c/My_TFG/src/app/tools/slas_C001/data/"

resultado=-1
id_ejecucion=-1 #Si es una ejecución automática, $1 = -1
id_ejecucion=$1 #Si es una ejecución manual,     $1 != -1
#En una ejecución manual el argumento debe coincidir con una entrada tabla
# bbdd_cumplimientos_sla.ctrl_estado_ejecucion introducida manualmente

#INICIALIZACION DE VARIABLES
py_calculo=$rutascripts"calculo_slas_C001.py"

fecha=`date`
datelog=`date +"%Y%m%d_%H%M%S"`

appName=sla_C001
contrato="C001"
sla="SLAC01_01"

#CREACION DEL LOG
logfile=$rutasdata"logs/"$appName"_"$datelog".log"

# CABECERA LOG
echo "#########################################################################################################"
echo "### Log de ejecucion para el SLA del contrato C001" >> $logfile
echo "### "$fecha >> $logfile
echo "#########################################################################################################"

chmod 755 $logfile
#EJECUCION DEL PROCESO
#Activamos el entorno virtual
. /mnt/c/My_TFG/src/app/tools/slas_C001/env_c001/bin/activate >> $logfile

#Guarda el resultado del print.
resultado=`python3 $rutascomun"inicio_ejecucion.py" $logfile $id_ejecucion $contrato $sla` >> $logfile

if [[ $resultado -gt 0 ]]; then
    python3 $py_calculo $logfile $resultado >> $logfile
fi

#Desactivamos el entorno virtual
deactivate
```

- Si es una ejecución automática, el proceso anterior habrá añadido una nueva entrada a la tabla 'ctrl_estado_ejecucion' y devolverá el identificador de la entrada correspondiente.

6. Por defecto habremos inicializado la variable 'resultado' a -1. Comprobamos que su valor sea positivo para poder continuar, si no, desactivamos el entorno y termina la ejecución.

7. En caso de ser un valor correcto, llamamos al último proceso que ejecutará el cálculo de cumplimientos, pasándole como argumento la ruta del log y la variable 'resultado' que contiene el id de de ejecución.

8. Por último, desactivamos el entorno virtual y terminamos la ejecución.

6.2.2. Proceso 2 : Inicio de Ejecución

1. Iniciamos el proceso importando las librerías que vamos a necesitar (pandas, datetime, calendar, sqlalchemy y sys) y los ficheros 'constantes.py' y 'funciones_comunes.py' que contienen nombres de constantes y funciones que serán de utilidad.

2. Almacenamos los argumentos con los que hemos hecho la llamada a la función (ruta log, id de ejecución, contrato y sla).

3. Inicializamos el resto de variables que vayamos a necesitar:

- Accedemos los atributos de conexión de la bbdd cuyos valores se encuentran en el fichero 'constantes.py' y los almacenamos en variables locales.
- Inicializamos las variables 'tipo_ejecucion' y 'modo_ejecucion' a 'MANUAL' y 'AUTOMATICA', respectivamente, ya que serán los valores por defecto de la ejecución.
- Guardamos dos listas con los tipos de contrato y tipos de sla existentes. Estas listas están guardadas en el fichero 'constantes.py' para que puedan actualizarse fácilmente en un futuro.
  tipos_contrato=['C001']
  tipos_sla=['SLAC01_01']
- Guardamos datos sobre la fecha de la ejecución del proceso: fecha completa, mes, año y semana actual.

4. Abrimos una conexión a la bbdd 'bbdd_cumplimientos_sla'.

5. Si se está realizando una ejecución automática (id_ejecucion==-1): Realizamos una consulta SQL sobre la tabla 'ctrl_estado_ejecucion' para obtener el último valor del campo 'id_ejecucion'. Será necesario conocer este para poder actualizar la tabla más adelante, añadiendo un nuevo registro con el valor leído +1.

6. Si estamos realizando una ejecución manual, abremos recibido el id de la ejecución como argumento. Realizamos una consulta sobre la misma tabla anterior para conocer los datos asociados a ese id como: contrato, sla, tipo de ejecución, modo de ejecución, mes, año y semana.

7. Necesitamos conocer el periodo para el que se realizará la ejecución. En una ejecución manual obtendremos estos datos de la consulta sql anterior, en caso contrario deberemos calcular el mes anterior a la fecha actual (teniendo en cuenta que si nos encontramos en enero, el mes anterior será diciembre del año anterior).

8. A continuación realizaremos las siguientes validaciones:

   - Comprobamos que los valores de los campos 'contrato' y 'sla' coincidan con alguno de los almacenados en la listas 'tipos_contrato' y 'tipos_sla' respectivamente.
   - Comprobamos que existan datos suficientes para la ejecución. Si se ha indicado que el tipo de ejecución es 'MENSUAL' se deberá haber indicado el campo 'mes' en la tabla y si la ejecución es 'SEMANAL', se deberá haber informado el campo 'semana'.
   - En caso de no se cumpla alguna de estas validaciones, actualizaremos la entrada de la tabla 'ctrl_estado_ejecucion' almacenando un mensaje de error e indicando que la ejecución ha fallado ('KO').

9. Además, deberemos comprobar que existan datos cargados en la BBDD posteriores a la fecha en la que queremos realizar el cálculo. Para ello calculamos el último día que entraría dentro de la ejecución. Para la ejecución mensual será el último día del mes a las 23:59:59.

10. Volvemos a abrir una conexión a la BBDD para consultar la fecha de la última carga en la tabla 'ctrl_estado_carga', para el fichero de hechos que corresponda con nuestro contrato (En este caso será aquella cuyo campo 'nombre_fichero' coincida con 'c001_tickets'). Si la fecha es mayor al último día del mes calculado, podrá realizarse el cálculo. En caso contrario actualizaremos la entrada de la tabla 'ctrl_estado_ejecucion' indicando que la ejecución ha fallado porque aun no existen datos.

11. Para finalizar, en caso de una ejecución automática, añadiremos una nueva entrada a la tabla 'ctrl_estado_ejecucion' con los siguientes valores:

- id_ejecucion: será el valor de la última entrada +1
- tipo_contrato: 'C001'
- tipo_sla: SLAC01_01
- fecha_ejecucion: fecha actual en formato %Y- %m- %d %h: %m: %s
- tipo_ejecucion: 'MENSUAL'
- mod_ejecucion: 'AUTOMATICA'
- anno, mes: valores calculados para el año y mes de la ejecución.
- resultado: 'WORK IN PROGRESS'

12. Imprimimos por pantalla el valor del campo 'id_ejecucion' para que el script lo recoja y almacene en una variable.

13. Terminamos la ejecución y devolvemos el control al script sh.

6.2.3. Proceso 3: Cálculos para el Contrato C001

1. Comienza el proceso e importamos las librerías necesarias y los ficheros de constantes y funciones que utilizaremos. Almacenamos los argumentos con los que hemos hecho la llamada a la función e inicializamos las variables que vayamos a necesitar.

2. Abrimos distintas conexiones con las BBDD para obtener las tablas que necesitaremos utilizar al realizar los cálculos y renombramos algunos campos de los dataframes para diferenciarlos y manejarlos mejor.

3. Obtenemos un dataframe con los tickets que se hayan solucionado o cerrado.

4. Para cada uno de los tickets, calcularemos las distintas fechas de resolución, restauración, reapertura y número de reaperturas que hayan tenido. Calculamos el año y mes de solución de cada uno para poder filtrar y quedarnos únicamente con aquellos que se encuentren dentro del periodo deseado. Para tickets sin reaperturas, el mes y año se calcularán sobre la fecha de la primera solución, mientras que para los tickets con reaperturas, será el mes y año de la ultima fecha de solución.

5. Calculamos también el día, semana y el día de la semana de cada ticket (Nos servirá para la visualización). También los tiempos de solución restauración de cada ticket.

6. Calculamos la tecnología definitiva de cada ticket y descartamos aquellos que no podamos utilizar por falta de información, ya sea por que no tengan tecnología asignada (N.A.), un nivel de severidad definido o aquellos cuyo elemento de red sea Genérico.

7. Calcularemos la prioridad a partir de la tecnología definitica que hemos calculado.

8. Para calcular el site de cada ticket necesitamos que el campo provincia esté informado. Realizaremos distintos cruces y cálculos para obtener la Provincia y el Site a partir de ella. Descartaremos aquellos tickets cuya provincia haya quedado sin informar.

9. Los tickets descartados en cada una de las partes se añadirán a un nuevo dataframe, incluyendo las razones de ello.

10. Tras calcular los campos a partir de los cuales podremos obtener los plazos de resolución y restauración de cada ticket, llegamos a la parte principal del proceso, comprobar si cada uno de los tickets cumple estos objetivos propuestos.

- A partir de la prioridad y el id del site, obtendremos los objetivos de resolución y restauración de cada ticket (figura 5.10)
- Un ticket habrá cumplido el objetivo de resolución, si su tiempo de primera solución es inferior al tiempo objetivo.
- Un ticket habrá cumplido el objetivo de restauración, si su tiempo de última solución es inferior al tiempo objetivo.

11. Añadimos distintas tablas a nuestro dataframe para almacenar el resultado de estas comprobaciones, identificando con 'SI' o 'NO' si se han cumplido o no cada uno de los objetivos.

12. Anonimizamos los datos sensibles que no deben ser incluidos en los resultados.

13. Preparamos los campos que van a ser cargados en la BBDD y en los ficheros excel. En especial aquellos que nos serán de utilidad para poder realizar la visualización e identificar fácilmente si se han cumplido los objetivos propuestos.

14. Por último cargamos los resultados en la BBDD y exportamos los ficheros en el directorio `C:\My_TFG\src\app\tools\slas_C001\data\output`

6.3. Visualización.

Como hemos visto, la visualización de datos se ha realizado con la herramienta de Power BI. Tras realizar distintas versiones, finalmente se ha creado un informe compuesto por tres páginas distintas, para distribuir la información y tener una visión más clara de esta.

Página 1

En primer lugar, tenemos una página cuyo objetivo es mostrarnos rápidamente la situación de cada uno de los KPIS definidos, para el mes seleccionado. Se ha decidido mostrar esta información mediante visualizaciones de KPI, ya que estos objetos permiten evaluar el estado actual de una métrica respecto a un objetivo definido. Apoyaremos esta visualización con unas gráficas de columnas agrupadas que nos permiten ver como están distribuidos los tickets para cada uno de los KPIs. En la parte superior se ha colocado una Tarjeta que nos muestra un recuento de la cantidad de tickets que se han gestionado.

Para el mes de Agosto de 2021, se han resuelto un total de 440 tickets, sin embargo no se han cumplido con éxito todos los KPIs.

- Los tickets con prioridad 1 y 2 asociados al KPI01 no han cumplido sus objetivos de tiempo de resolución. De 402 tickets, 10 de ellos no han superado dicho tiempo, por lo que el porcentaje de cumplimiento de dicho KPI ha sido 97,51 % en lugar del 99 % deseado.

- Para ese mismo objetivo de resolución, pero esta vez tickets con prioridades 3,4 y 5 se ha cumplido el KPI al 100 %, superando el 97 % esperado.

- En cuando al KPI03, cantidad de tickets solucionados en una primera interacción, no se ha superado el objetivo con una diferencia de 2,87 % (92,27 %/95 %). De los 440 tickets, 34 de hechos han tenido alguna reapertura, por lo tanto no han cumplido este objetivo.

- Finalmente, el KPI04 relacionado con el tiempo de solución (restauración) de tickets con reaperturas, tampoco se ha cumplido. De los 34 tickets con reapertura, 5 de ellos no superado los plazos marcados, por tanto el KPI se ha superado en un 85,29 %, un 5,23 % menos que el objetivo marcado.

Páginas 2 y 3

Las siguientes páginas están compuestas por las mismas visualizaciones, sin embargo, cada una representa un tipo de información distinta. La segunda página permite visualizar características particulares de aquellos tickets que No han cumplido los objetivos de tiempo de resolución(primera solución), mientras que la tercera mostrará esto mismo para aquellos que no han cumplido el objetivo de restauración (Tiempo solución reapertura).

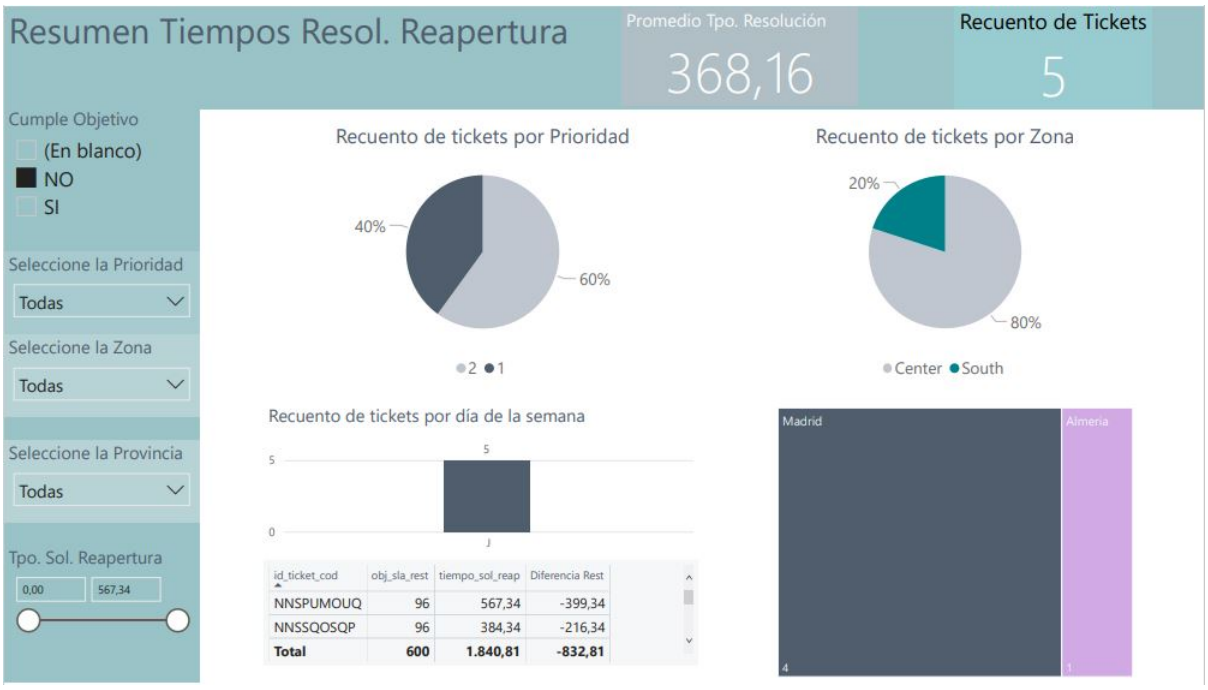
A diferencia de la anterior, en estas páginas se han añadido filtros para poder seleccionar los datos que más nos interesen. Uno de ellos servirá para filtrar los tickets que cumplan o no los pautas de resolución establecidas y vendrá seleccionado por defecto para mostrar directamente aquellos que no lo hagan. Otros filtros añadidos han sido desplegables para poder seleccionar los tickets con una determinada prioridad o situados en una zona o provincia concreta. En cuanto a los gráficos utilizados, se han añadido varios gráficos circulares y un Treemap para mostrar la información mencionada, además de un gráfico de columnas agrupadas para visualizar el volumen de tickets por día de la semana y una tabla con varias columnas para mostrar la diferencia entre el tiempo de solución objetivo y el tiempo real de cada ticket.

Como hemos visto en la primera página, 10 de los 440 tickets no han cumplido su objetivo de resolución.

- De los 10 tickets, 6 de ellos tienen una prioridad 2 y 4 prioridad 1.
- El 50 % de esos tickets pertenecen a la zona centro, 3 de ellos a Madrid, 1 a Toledo y 1 a Las Palmas.
- El 40 % de esos tickets pertenecen a la zona Este, 2 de ellos a Barcelona, 1 a Valencia y otro a Baleares.
- El 10 % finalmente se encuentra en Palencia y pertenece a la zona Centro.
- En cuanto al día de la semana en el que se concentran más incidencias, los Martes y Miércoles agrupan 4 y 3 tickets respectivamente.

De igual forma, anteriormente hemos visto que 5 de los 34 tickets con reaperturas no han cumplido sus respectivos objetivos de reapertura.

- Los 5 tickets tienen prioridades 1 y 2.
- El 80 %, 4 tickets se concentran en Madrid (Zona Centro) mientras que el restante se localiza en Almería (Zona Sur).
- En cuanto al volumen por día de la semana, vemos que el Jueves se han resuelto el 100 % de dichos tickets.



Capítulo 7

Pruebas

En este apartado se describen las pruebas realizadas para comprobar el correcto funcionamiento del sistema. Las pruebas se han dividido en *Pruebas de Carga*, que comprobarán el proceso de NIFI y *Pruebas de Cálculo*, que comprobarán el funcionamiento del proceso ETL de cálculo de cumplimientos. A continuación se muestra un resumen y los resultados esperados y obtenidos tras realizarlas.

7.1. Pruebas de Carga

Prueba 1.1 - Carga inicial de datos en la BBDD	
Descripción	En esta prueba se simulará la primera carga del fichero de hechos para el contrato 'C001'.
Acción	Se copiará el fichero 'c001_tickets.csv' en el directorio C:\My_TFG\src\app\tools\comunes\data\mailbox para poder comenzar el proceso NIFI.
Antecedentes	La tabla *fact_tickets_c001* se encuentra vacía y el registro de la tabla *ctrl_estado_carga* correspondiente al fichero 'c001_tickets' tendrá sus campos a NULL salvo 'estado_carga_id','nombre_fichero','tipo_contrato' y 'tipo_sla'
Resultados Esperados	- El fichero csv se habrá movido al directorio C:\My_TFG\src\app\tools\comunes\data\processed - Los datos del fichero csv se habrán cargado por completo en la tabla *fact_tickets_c001.* - Se habrán actualizado los campos de la tabla *ctrl_estado_carga* para el registro correspondiente, indicando la fecha de la ejecución y la ruta donde se ha depositado el fichero. - Se habrá insertado un 'OK' en el campo *resultado* de la tabla *ctrl_estado_carga* indicando que el proceso ha finalizado correctamente.

Tabla 7.1: Prueba 1.1

Prueba 1.2 - Actualización de datos en la BBDD	
Descripción	Se simulará la carga de un fichero cuya BBDD ya contiene datos de una carga anterior.
Acción	Se copiará el fichero 'c001_tickets.csv' en el directorio C:\My_TFG\src\app\tools\comunes\data\mailbox para poder comenzar el proceso NIFI.
Antecedentes	- La tabla *dim_prov_zona_id* contiene datos sin actualizar y - El registro de la tabla *ctrl_estado_carga* correspondiente al fichero tendrá todos sus campos informados con los datos de la última carga.
Resultados Esperados	- Se eliminarán los registros anteriores de la tabla *dim_prov_zona_id* y se cargarán los nuevos. el campo 'fecha_modificacion' se actualzizará con la fecha actual - Los campos de la tabla *ctrl_estado_carga* para el registro correspondiente se habrán actualizado, y se habrá insertado un 'OK' en el campo *resultado*

Tabla 7.2: Prueba 1.2

Prueba 1.3 - Carga Fallida de datos en la BBDD	
Descripción	Se forzará un error en la carga de datos para comprobar que se registra correctamente. Los posibles fallos que pueden darse son: 1. Tipo de fichero erróneo. El fichero no es de tipo csv. 2. Algún metadato de la tabla *ctrl_info_fichero* es incorrecto y no pueden cargarse los datos en la tabla correcta. 3. El fichero cargado no tiene el formato adecuado.
Acción	Se realizarán las siguientes modificaciones sobre los ficheros 'perfil_tecnicos.csv' , 'sev_prio.csv' y 'zona_prov.csv': 1. Para el primero, se cambiará el formato del fichero a xlsx. 2. Para el segundo, se modificará el valor del campo 'tabla_bbdd' en la tabla *ctrl_info_fichero*. 3. En el último fichero, se cambiará el delimitador ';' por ','.
Resultados Esperados	En ninguno de los casos se habrán cargado los datos del fichero. Se habrán modificado sus correspondientes registros en la tabla *ctrl_estado_carga*. Se habrá insertado un 'KO' en el campo 'resultado' y un mensaje de error en el campo 'log', dependiendo del tipo de error: 1. En el primer caso el error será 'EL FORMATO DE FICHERO NO ES EL ESPERADO (CSV)'. 2. En los dos últimos casos: 'ERROR AL OPERAR CON LA BBDD'.

Tabla 7.3: Prueba 1.3

7.2. Pruebas ETL - Cálculo cumplimientos

Prueba 2.1 - Ejecución Manual con información errónea o incompleta.	
Descripción	Se simulará una ejecución manual del proceso ETL, pero la tupla de datos introducida en la tabla *ctrl_estado_ejecucion* es incorrecta, le faltan campos o estos son erróneos. Son 5 los campos que deben rellenarse son: tipo_contrato', 'tipo_sla', 'anno' y 'mes' o 'semana' en función del tipo de ejecución (será mensual por defecto).'
Acción	Se insertarán los siguientes valores en los campos mencionados anteriormente: 1. valor 'ABCDE' en el campo 'tipo_contrato'. 2. valor 'ABCDE' en el campo 'tipo_sla'. 3. valor NULL en el campo 'mes'.
Resultados Esperados	En los tres casos, la ejecución finalizará tras comprobar que los campos no son correctos. Se habrán actualizado los datos de los registros correspondientes en la tabla *ctrl_estado_ejecucion*. Se habrá insertado un 'KO' en el campo 'resultado' y un mensaje de error en el campo 'log', dependiendo del tipo de error: 1. En primer y segundo caso el error será 'EL TIPO DE CONTRATO O DE SLA NO ES VÁLIDO' 2. En el último caso será: 'FALTAN DATOS PARA EL TIPO DE EJECUCIÓN (MENSUAL O SEMANAL)'.

Tabla 7.4: Prueba 2.1

Prueba 2.2 - Ejecución Automática sin fichero cargado.	
Descripción	Se simulará una ejecución automática del proceso ETL, sin haber cargado previamente el fichero de tickets para el mes anterior.
Resultados Esperados	La ejecución finalizará sin haber añadido un nuevo registro en la tabla *ctrl_estado_ejecucion*

Tabla 7.5: Prueba 2.2

Prueba 2.3 - Ejecución Automática sin tickets .	
Descripción	Se simulará una ejecución automática del proceso ETL, habiendo cargado el fichero del mes anterior pero sin tickets de incidencia.
Resultados Esperados	Se habrá añadido una nueva entrada en la tabla *ctrl_estado_ejecucion*. La ejecución finalizará sin realizar ningún cálculo y se habrá insertado un 'KO' en el campo 'resultado' y el mensaje de error 'NO EXISTEN DATOS PARA LA FECHA INDICADA' en el campo 'log'

Tabla 7.6: Prueba 2.3

Prueba 2.1 - Ejecución Correcta	
Descripción	Se simulará una ejecución del proceso ETL para el que existen datos cargados en el periodo deseado.
Resultados Esperados	La tabla *ctrl_estado_ejecucion* tendrá una nueva tupla cuyo campo 'resultado' será 'OK', indicando que el cálculo ha finalizado con éxito. Se habrán cargado los resultados del cálculo en las tablas con prefijo *out_* de la BBDD. Se habrán descargado los resultados en el directorio C:\My_TFG\src\app\tools\slas_C001\data\output, en distintos ficheros .xlsx

Tabla 7.7: Prueba 2.4

Capítulo 8

Conclusiones y Trabajo Futuro

Tras finalizar este proyecto puedo afirmar que se han conseguido los objetivos planteados al inicio de este, no solo a nivel técnico o de investigación sino también se han conseguido alcanzar objetivos personales de aprendizaje.

En lo personal, me ha permitido poner en práctica conceptos adquiridos durante la carrera que han servido para sacar adelante un proyecto de fin de carrera, en cuanto a la planificación, metodología o diseño de este. Además de haber descubierto y haberme familiarizado con herramientas nuevas, también he podido ampliar mis habilidades y conocimientos técnicos sobre otras como Python o Power BI, con las que ya había trabajado muy brevemente. Con Python he descubierto un lenguaje de programación muy versátil y esencial para Data Science y Big Data, un tema que me interesa y en el que quiero seguir formándome, al igual que con Business Intelligence.

Gracias a este proyecto, he sido consciente de la importancia de una buena gestión y planificación inicial. Es muy importante definir correctamente los requisitos y objetivos que queremos conseguir, ya que ayudarán a orientar correctamente el proyecto, facilitando las tareas de diseño e implementación. Pero sobre todo, es imprescindible realizar una buena estimación de tiempos y gestión de riesgos previa. Pese a conocer esto y haber considerado los riesgos y sus posibles retrasos en el proyecto, la fecha de finalización prevista inicialmente se ha visto afectada por diversos motivos, ya sea por enfermedad, bajo rendimiento o la necesidad de emplear más tiempo en alguna de las tareas planificadas. A pesar de ello y dada la naturaleza del proyecto estos retrasos no han supuesto grandes consecuencias.

En cuanto al propio proyecto se refiere, ha quedado latente la importancia y necesidad de incorporar estrategias de análisis y visualización de datos a cualquier negocio, para poder sacarles el mayor partido posible y así aumentar el rendimiento y los beneficios de la empresa. Por otro lado, al haber implementado un sistema automatizado que permita calcular el cálculo de cumplimientos de SLAs, se está reduciendo el tiempo empleado en realizar estas tareas de forma manual y minimizando los errores humanos que pueda conllevar.

A pesar de haber conseguido los objetivos propuestos, aún queda trabajo por hacer y mejoras que se podrían realizar sobre el sistema. Como se comentó en un principio, el proyecto se ha diseñado pensando en la posibilidad de poder aplicar fácilmente los procesos implementados sobre distintos contratos SLAs y distintas periodicidades, por lo que una posible mejora sería implementar la posibilidad de realizar ejecuciones semanales, donde se sacarán los tickets resueltos de una semana hacia atrás.

En cuestiones de rendimiento, existe una función en el proceso 'calculo_slas_c001.py', FechasSolCierre(), cuyos tiempos de ejecución serán cada vez mayores a medida que crezca el tamaño de la tabla 'fact_tickets_c001'. Esta función es la encargada de calcular los tiempos de solución de cada ticket, pero lo hace sobre el conjunto total de esta tabla independientemente de que estos se encuentren en el rango deseado o no, por lo que a medida que esta crezca tendrá peor rendimiento. Por ello, se deberá buscar una alternativa o solución que permita descartar con antelación los tickets de los cuales se deben calcular los tiempos de solución o bien reformular dicha función. Para comprobar el rendimiento del proceso 'calculo_slas_c001.py' se han realizado 5 ejecuciones para 3 volúmenes de tickets distintos, 2625,4621 y 7312.

- Tiempo medio para 2625 registros: 0,57387 segundos
- Tiempo medio para 4621 registros: 1,28995 segundos
- Tiempo medio para 7312 registros: 2,21325 segundos

Si hablamos de la visualización, existe un gran margen de mejoría ya que los informes creados se han hecho bajo un criterio personal y solo incluyen información sobre el mes actual y de esta manera no podemos observar patrones entre distintos meses o años. Además, los informes creados podrán variar considerablemente en función de quien sea el usuario objetivo, la información que se quiera visualizar y la manera en que se quiera hacer. Pese a ello, una posible mejora que puede interesar en cualquier situación sería incluir un histórico o bien obtener una imagen global de la situación de la empresa durante un año completo. Esto será útil para poder comprobar si se están mejorando los resultados de un mes a otro o incluso, si se tienen datos suficientes, de un año a otro.

Para finalizar, la última propuesta está relacionada con la herramienta NIFI. Esta tiene un gran potencial y las posibilidades que pueden implementarse con ella son infinitas, pero a pesar de disponer de manuales sobre cada uno de sus procesadores, existen pocos ejemplos prácticos que puedan consultarse. Por ello, se pretende investigar más sobre esta herramienta y poder ampliar las funcionalidades del sistema con ella, en cuanto al tratamiento de flujos de datos se refiere.

Referencias

[1] 4 consejos para crear una matriz de riesgos, Ultimo acceso: abril 2022. URL: https://www.escuelaeuropeaexcelencia.com/2018/10/4-consejos-para-crear-una-matriz-de-riesgos/.

[2] 7 riesgos comunes de un proyecto y cómo prevenirlos, Ultimo acceso: abril 2022. URL: https://asana.com/es/resources/project-risks.

[3] Anaconda distribution: La suite más completa para la ciencia de datos con python, Ultimo acceso: mayo 2022. URL: https://blog.desdelinux.net/ciencia-de-datos-con-python/.

[4] Apache nifi, una plataforma de logística de datos integrados en tiempo real y de procesamiento de eventos sencillos, Ultimo acceso: abril 2022. URL: https://es.cloudera.com/products/open-source/apache-hadoop/apache-nifi.html.

[5] Ciencia de datos: Anaconda pythonn, Ultimo acceso: mayo 2022. URL: https://www.tokioschool.com/noticias/ciencia-datos-anaconda-python/.

[6] Cómo evaluar las consecuencias y la probabilidad en el análisis de riesgos iso 27001, Ultimo acceso: abril 2022. URL: https://www.escuelaeuropeaexcelencia.com/2019/03/como-evaluar-las-consecuencias-y-la-probabilidad-en-el-analisis-de-riesgos-iso-27001/ /.

[7] Diferencia entre un kpi y un acuerdo de nivel de servicio, Ultimo acceso: marzo 2022. URL: https://tudashboard.com/diferencia-entre-un-kpi-y-un-acuerdo-de-nivel-de-servicio/.

[8] Migración de datos; ¿cómo realizarla y qué problemas surgen?, Ultimo acceso: marzo 2022. URL: https://www.ticportal.es/glosario-tic/migracion-datos.

[9] Monitorización para medir el rendimiento de infraestructuras, Ultimo acceso: marzo 2022. URL: https://blog.a3sec.com/monitorizaci%C3%B3n-rendimiento-infraestructuras#:~:text=Los%20SLAs%20son%20m%C3%A9tricas%20que,los%20est%C3%A1ndares%20de%20servicio%20definidos.

[10] Procesos etl: cómo obtener valor de los datos, Ultimo acceso: marzo 2022. URL: https://www.cognodata.com/procesos-etl/.

[11] Qué es la gestión de riesgos y cómo aplicarla a tu proyecto en solo 6 pasos, Ultimo acceso: abril 2022. URL: https://asana.com/es/resources/project-risk-management-process.

[12] Salario medio para ingeniero informático en españa, 2022, Ultimo acceso: mayo 2022. URL: https://es.talent.com/salary?job=ingeniero+inform%C3%A1tico#:~:text=El%20salario%20ingeniero%20inform%C3%A1tico%20promedio,hasta%20%E2%82%AC%2030.000%20al%20a%C3%B1o.

[13] Sla informática: lo que tienes que saber, Ultimo acceso: marzo 2022. URL: https://apser.es/sla-informatica-lo-que-tienes-que-saber/#:~:text=Son%20las%20siglas%20en%20ingl%C3%A9s,la%20calidad%20del%20servicio%20prestado.

[14] Tabla de amortización para sociedades y autónomos en estimación directa, Ultimo acceso: abril 2022. URL: https://www.gabilos.com/webcontable/amortizacion/estimacion_directa_normal.html.

[15] Tabla de amortización simplificada, Ultimo acceso: abril 2022. URL: https://sede.agenciatributaria.gob.es/Sede/ayuda/manuales-videos-folletos/manuales-practicos/folleto-actividades-economicas/3-impuesto-sobre-renta-personas-fisicas/3_5-estimacion-directa-simplificada/3_5_4-tabla-amortizacion-simplificada.html.

[16] Tarifa de primas por contingencias profesionales vigentes para 20222, Ultimo acceso: abril 2022. URL: https://www.iberley.es/temas/tarifa-primas-contingencias-profesionales-accidente-trabajo-enfermedad-profesional-2022-6571

[17] ¿cuánto se gana en españa, como ingeniero/a informático/a?a, Ultimo acceso: mayo 2022. URL: https://es.indeed.com/career/ingeniero-inform%C3%A1tico/salaries?from=careeradvice-ES.

[18] ¿cómo definir y qué incluir en un acuerdo de nivel de servicio (ans o sla)?, Ultimo acceso: marzo 2022. URL: https://www.ambit-bst.com/blog/c%C3%B3mo-definir-y-qu%C3%A9-incluir-en-un-acuerdo-de-nivel-de-servicio-ans-o-sla.

[19] ¿qué es apache nifi? aspectos clave, Ultimo acceso: mayo 2022. URL: https://aprenderbigdata.com/introduccion-apache-nifi/.

[20] ¿qué es microsoft project?, Ultimo acceso: mayo 2022. URL: https://aglaia.es/blog/office-365/que-es-microsoft-project/#:~:text=Microsoft%20Project%20(MSP)%20es%20la,de%20calendarizar%20y%20organizar%20tareas.

[21] ¿qué es power bi?, Ultimo acceso: mayo 2022. URL: https://www2.deloitte.com/es/es/pages/technology/articles/que-es-power-bi.html.

[22] ¿qué es un service level agreement (sla)?, Ultimo acceso: marzo 2022. URL: https://www.servicetonic.com/es/service-desk/que-es-un-sla/.

[23] Enrique Arriols. Procesos etl: definición, desarrollo y aplicaciones, Ultimo acceso: marzo 2022. URL: https://blog.mdcloud.es/procesos-etl-definicion-desarrollo-aplicaciones/.

[24] Arsys. Qué es sla y cuáles son sus componentes clave, Ultimo acceso: marzo 2022. URL: https://www.arsys.es/blog/que-es-sla-y-cuales-son-sus-componentes-clave#Componentes_del_SLA.

[25] Douglas da Silva. ¿qué es sistema de ticketing?, Ultimo acceso: marzo 2022. URL: https://www.zendesk.com.mx/blog/ticketing/.

[26] Ministerio de Educación y Formación Profesional, Ultimo acceso: marzo 2022. URL: https://www.educacionyfp.gob.es/italia/dam/jcr:b53864d2-65a3-4526-abf4-61ef02f5be34/el-sistema-universitario-espa-ol2.pdf.

[27] Toni Exposito Escobar. ¿qué es un etl? ¿cuál es su importancia?, Ultimo acceso: marzo 2022. URL: https://www.linkedin.com/pulse/qu%C3%A9-es-un-etl-cu%C3%A1l-su-importancia-juan-antonio-exposito-escobar/?originalSubdomain=es.

[28] European Knowledge Center for Information Technology. Sistema de ticketing, Ultimo acceso: marzo 2022. URL: https://www.ticportal.es/glosario-tic/sistema-ticketing.

[29] European Knowledge Center for Information Technology. ¿qué beneficios de una herramienta de ticketing existen?, Ultimo acceso: marzo 2022. URL: https://www.3cx.es/helpdesk/herramienta-de-ticketing/.

[30] Paul Kirvan. Por qué el cumplimiento de sla debe estar en la mente de los líderes de ti, Ultimo acceso: marzo 2022. URL: https://www.computerweekly.com/es/consejo/Por-que-el-cumplimiento-de-SLA-debe-estar-en-la-mente-de-los-lideres-de-TI.

Anexo A

Instalación de Entornos Virtuales

Tras descargar e instalar el software Anaconda, debemos crear un entorno de trabajo a partir del cual escribiremos nuestros procesos de Python. Los entornos virtuales permiten que, para cada proyecto que desarrollemos, podamos mantener una instalación de paquetes aislados de la instalación principal que tenemos en nuestro sistema. Además de crear estos entornos en Anaconda, también ha sido necesario hacerlo en nuestra máquina local utilizando el subsistema windows de linux.

Las librerías y versiones utilizadas para este proyecto han sido las siguientes:

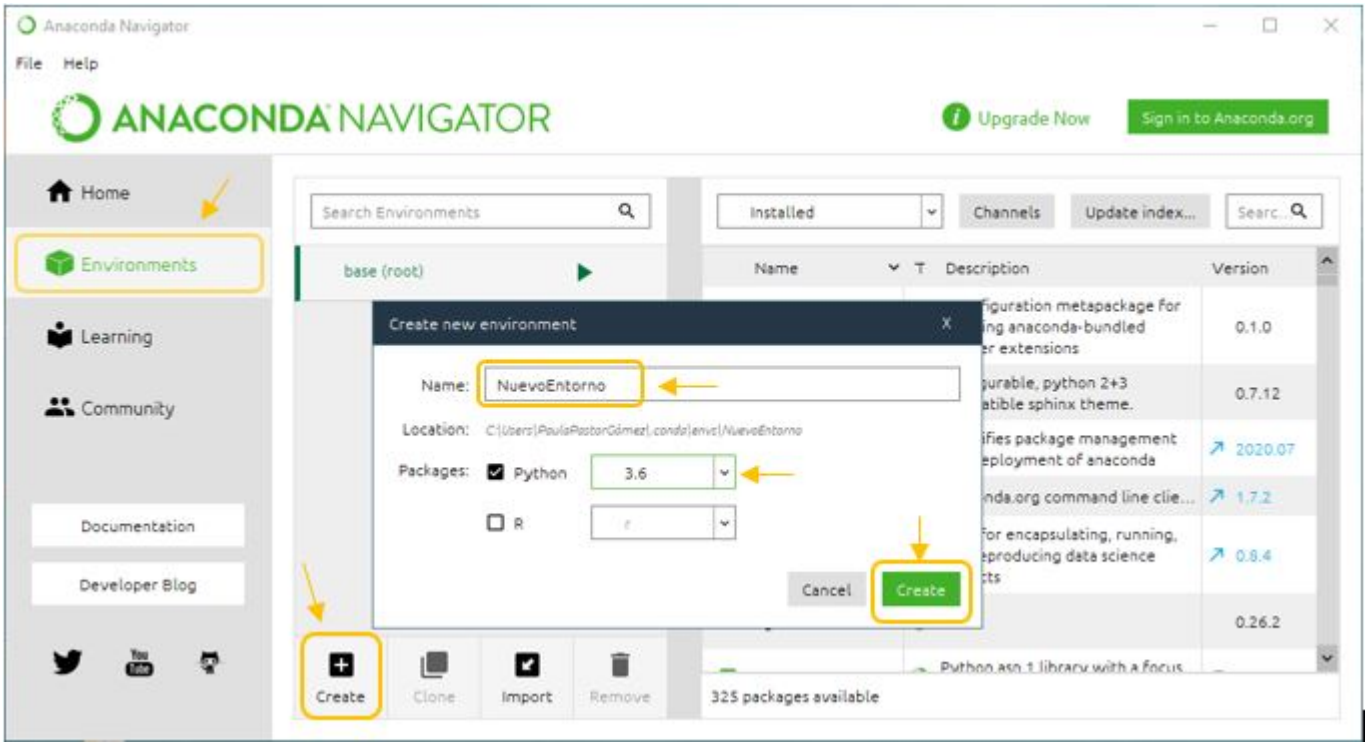
Python=3.6.9
openpyxl=2.4.8
pandas=1.1.1
sqlalchemy=1.3.16
pymysql=0.9.3
numpy=1.19.1

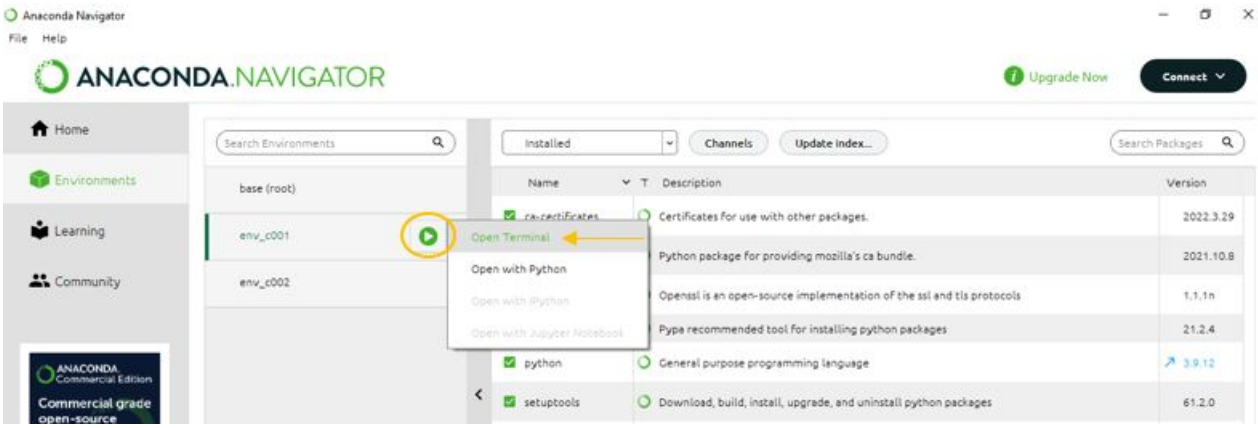
A.1. Creación de Entornos en Anaconda

En este apartado del anexo se muestra como crear entornos virtuales en Python con Anaconda, uno para cada proyecto o propósito. Antes de comenzar, se recomienda iniciar la aplicación Anaconda Navigator con permisos de Administrador, ya que se han observado problemas al crear nuevos entornos y/o instalar versiones.

1. En el menú izquierdo, seleccionamos la opción 'Enviroments'. Después sobre la opción 'Create +' para añadir un nuevo entorno. En la ventana emergente, indicamos el nombre que queremos darle, la versión de python que queremos instalar y pulsamos el botón 'Create' para finalizar.

2. Una vez creado, abrimos el terminal del nuevo entorno para instalar las versiones de las librerías que necesitemos. Hacemos click sobre el botón de 'Play' y seleccionamos 'Open Terminal'.





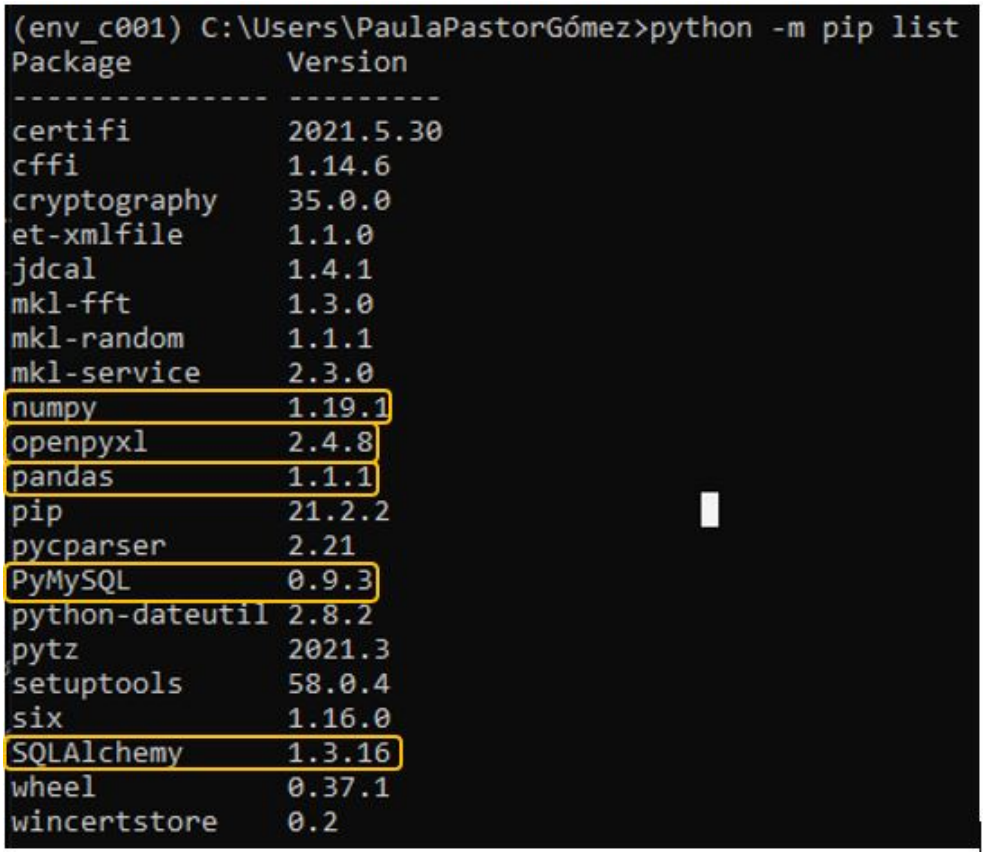
3. Para instalar las versiones que deseamos debemos hacerlo con el comando:

   conda install package_name
   conda install package_name=version

```
(env_c001) C:\Users\PaulaPastorGómez>conda install Python=3.6.9
```

4. Por último, comprobamos que se han instalado correctamente los paquetes y sus versiones, utilizando los siguientes comandos:

   python -m pip list
   conda list
   python –version



A.2. Creación de Entornos Virtuales en WSL

En este apartado del anexo se explican los comandos utilizados para crear entornos virtuales de Python en nuestra máquina local, utilizando el subsistema de windows para linux (WSL). Para este proyecto en particular crearemos un entorno llamado 'env_c001' en la siguiente ruta:

- (Windows) C:\My_TFG\src\app\tools\slas_C001\
- (WSL) /mnt/c/My_TFG/src/app/tools/slas_C001/

1. En primer lugar, instalamos los paquetes pip (ayuda a instalar, desinstalar y actualizar paquetes) y venv (para crear entornos virtuales).
   sudo apt update
   sudo apt upgrade
   sudo apt install python3-pip
   sudo apt install python3-venv

2. A continuación, desde la carpeta donde queremos crear el entorno, ejecutamos el siguiente comando:
   python3 -m venv nombre_entorno

   ```
   paupast@DESKTOP-31VRN9P:/mnt/c/My_TFG/src/app/tools/slas_C001$ python3 -m venv env_c001
   ```

   Este habrá creado un directorio llamado 'env_c001' que contendrá pip, intérprete, scripts y librerías.

3. Para instalar las versiones de las librerías creamos un archivo txt (requirements_c001.txt) como se muestra en la imagen A.1 y lo almacenamos en el directorio del entorno. A continuación deberemos activar este para poder realizar la instalación de una sola vez:
   source nombre_entorno/bin/activate (Activará el entorno)
   pip install -r requirements_c001.txt (Instalará las versiones).

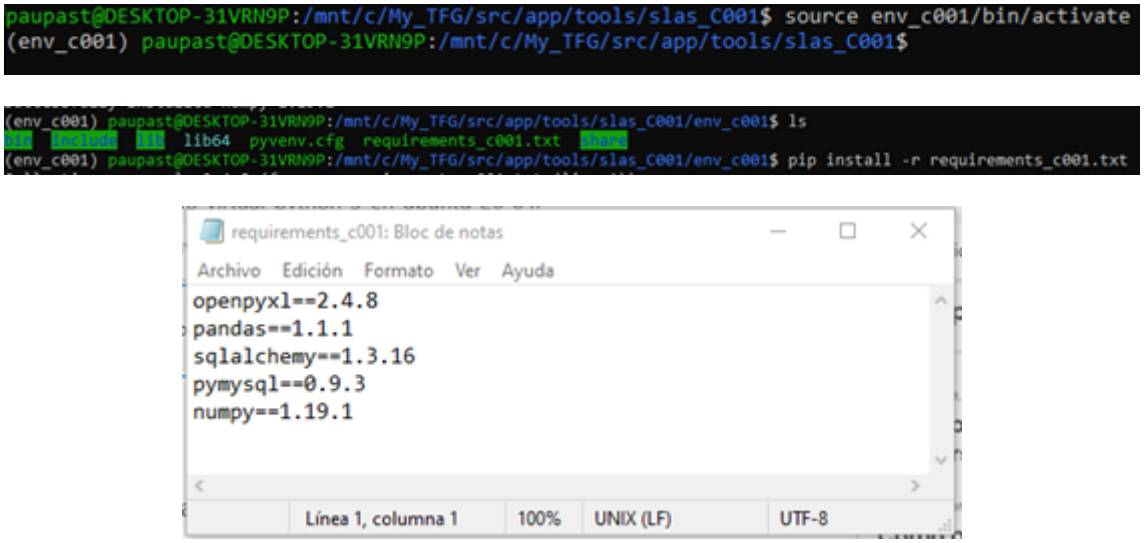
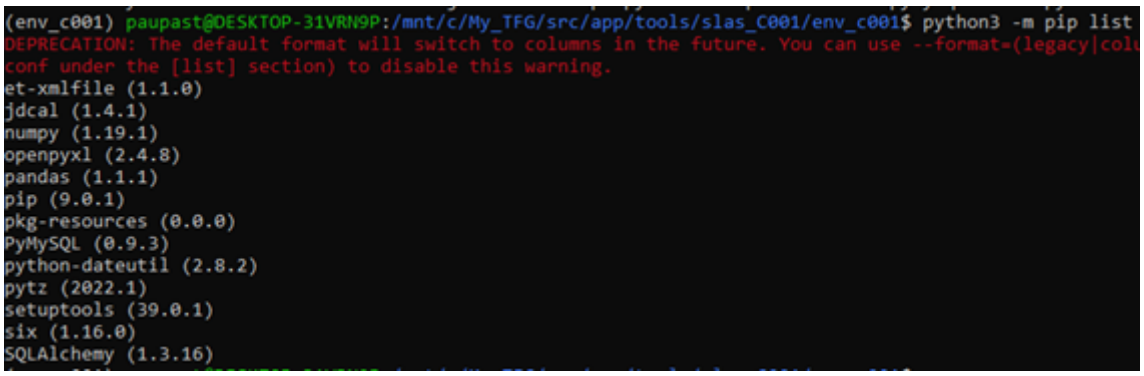


Figura A.1: Contenido fichero requirements

4. Para comprobar que se han instalado correctamente todas las versiones, ejecutamos el comando: *python -m pip install*



5. Para finalizar salimos del entorno con el comando *Deactivate*

Anexo B

Manuales de Uso y Despliegue

B.1. Manual Usuario para el Proceso 3

En este anexo se describirán detalladamente los pasos que se han llevado a cabo para implementar el proceso *'calculo_slas_c001.py'*.

1. Iniciamos el proceso importando las librerías que vamos a necesitar (pandas, numpy, datetime, sqlalchemy y sys) y los ficheros 'constantes.py' y 'funciones_comunes.py' que contienen nombres de constantes y funciones que serán de utilidad.

2. Almacenamos los argumentos con los que hemos hecho la llamada a la función (ruta log, id de ejecución).

3. Inicializamos el resto de variables que vayamos a necesitar:

   - Conexión y nombre de las bases de datos a las que accederemos.
   - Fecha de carga (para controlar la fecha en la que se generaron los resultados).
   - Ruta donde se almacenarán los ficheros Excel con los resultados.

4. A continuación abriremos distitnas conexiones con las BBDD para obtener todos los datos que necesitaremos utilizar para realizar la ejecución.

5. Conexión con la BBDD 'bbdd_tickets':

   - Consultamos la tabla 'dim_estado_tickets' para obtener una dataframe con las posibles variantes de los estados 'SOLUCIONADO' y 'CERRADO para los tickets de tipo 'Red'. Convertimos este df (dataframe) en una lista para poder utilizarla en una nueva consulta sql.

6. Conexión con la BBDD 'bbdd_cumplimientos_sla':

   - Consultamos la tabla 'ctrl_estado_ejecucion' para obtener los metadatos asociados al id pasado como argumento: tipo de contrato, tipo de sla, tipo de ejecución, modo de ejecución, mes y año.
   - Consultamos la tabla fact_tickets_c001 para obtener y almacenar en un dataframe los datos de tickets con los que realizaremos el resto del procedimiento. La consulta será sobre aquellos cuyo campo 'EstadoTicket' coincida con alguno de los de la lista obtenida en el punto 5.
   - Leemos el resto de tablas de dimensiones y las almacenamos en distintos dataframes(df).

7. Conexión con la BBDD 'bbdd_inventario':

- Realizamos un cruce entre los campos de categoría de las tablas 'cdr_equipment' y 'familias_tech' para obtener un df con los campos: 'Name', 'Site ID' y 'technology'.
- Leemos los campos 'SiteName', 'Province', 'SiteID' de la tabla 'cdr_sites' y los campos 'elemento_red' y 'provincia' de la tabla 'dim_emp_fijo' que nos servirán de ayuda para informar el campo Provincia de los tickets.

8. Renombramos algunos campos de los dataframes para diferenciarlos y manejarlos mejor.
9. Obtenemos una lista con ids de los tickets leídos.
10. Volvemos a abrir una conexión con la BBDD 'bbdd_tickets' y leemos los campos de la tabla amd_tickets_estados únicamente para los tickets que se encuentren en la lista que hemos obtenido en el punto 9.
11. ***Cálculo de Fechas de solucion y Reaperturas***
    - A partir del df obtenido en el punto 10, la lista de posibles estados 'SOLUCIONADO' y 'CERRADO'(punto 5), y la lista de tickets (punto 9), realizaremos una llamada a la función 'FechasSolCierre' para obtener las fechas de solucionado, reapertura, última solución, cantidad de reaperturas e identificador del técnico de cada uno de los tickets. La función nos devolverá un dataframe con estos datos.
    - Obtenemos el año y mes de solución de cada ticket para filtrar y quedarnos únicamente con aquellos que se encuentren dentro del periodo deseado. Para tickets sin reaperturas, obtendremos el mes y año de la fecha de la primera solución, mientras que para los tickets con reaperturas, será el mes y año de la ultima fecha de solución.
    - Obtenemos también el día y la semana de solución. Calculamos el día de la semana de cada ticket (Nos servirá para la visualización).
    - Unimos el dataframe original de los tickets con el nuevo que contiene las fechas de solución filtradas.
    - Calculamos los tiempos de solución y restauración de cada ticket. Los tickets sin reaperturas no tendrán tiempo de restauración.
12. ***Cálculo de Tenología Definitiva***
    - Convertimos a mayúsculas los campos que contienen identificadores de los elementos de red del df con los tickets y el que hemos obtenido de la tabla 'cdr_equipment', para realizar un cruce sobre dichos campos ('ElementodeRed' y 'Name' respectivamente) y obtener el campo 'technology'.
    - Convertimos este campo a mayúscula y lo almacenamos en uno nuevo que llamaremos 'tecnologia_def'. A continuación realizaremos comprobaciones para obtener los valores definitivos de tecnología que sean 'FIJO', 'MOVIL' o 'AMBAS'.
    - Si el campo 'tecnologia_def' está vacío lo sustituimos por 'N.A.'
    - Si el campo 'Tecnología' (original tabla hechos) contiene 'CABLE', el valor del campo 'tecnologia_def' será 'FIJO'.
    - Si el campo 'Tecnología' es 'AMBAS' y la 'tecnologia_def' es 'N.A.', el valor del campo 'tecnologia_def' será 'AMBAS'.
    - Si los valores del campo 'tecnologia_def' son 'TX' o 'CENTRO CORE' se traducirán como 'MOVIL' y 'FIJO' respectivamente.

- Borramos del dataframe las columnas que no nos vayan a servir más ('Tecnologia','Name','technology').

13. En este punto podremos descartar ciertos tickets que no podremos utilizar por falta de información. Serán aquellos que no tengan tecnología asignada (N.A.), un nivel de severidad definido o aquellos cuyo elemento de red sea Genérico. Crearemos un nuevo dataframe con estos descartes, añadiendo las razones para ello.

14. ***Cálculo de Prioridad***

    - Para calcular la prioridad una vez obtenida la tecnología definitiva, sólo es necesario unir el dataframe con el obtenido al leer la dimensión 'dim_sev_prio'.
    - Tras el cruce es posible que queden algunos campos de prioridad sin resolver ya que la tabla de dimensión solo contiene las tecnologías 'Fijo' y 'Movil'. Para aquellos tickets cuya tecnología definitiva sea 'AMBAS', llamaremos a una función que calcule la prioridad más alta entre los valores de ambas tecnologías (Fijo y Móvil), para la misma severidad.

15. ***Cálculo de Provincia y Site***

    - Comenzaremos realizando dos cruces sobre los dataframes de las tablas 'cdr_sites' y 'dim_emp_fijo' para obtener los campos con provincias correspondientes.
    - Renombramos estos para distinguir cual es su origen.
    - Si el campo original de provincia (origen en la tabla de hechos) está vacío, le asignaremos el valor que tuviera la provincia de la tabla 'cdr_sites'.
    - Para los tickets que aun tengan este campo vacío y cuya tecnología fuera 'Fijo', se les asignará la provincia correspondiente de la tabla 'dim_emp_fijo'
    - Como última opción se comprobaría el campo 'Datosdelticket_Causa' (original en la tabla hechos) para obtener detalles sobre la Provincia. En este campo, la provincia vendrá seguida del string 'Provincia' o 'Provincia//', así que lo primero que haremos será eliminar los caracteres '//'. Después iteraremos sobre cada fila del dataframe y separaremos el campo en una lista (separando los strings por espacios). El primer elemento será el string 'Provincia' y el segundo la provincia en sí, por lo que almacenaremos este dato en el campo de la Provincia.
    - Tras esto, descartaremos los tickets que no tengan provincias y los añadiremos al dataframe de descartes.
    - Para obtener el Site, convertiremos a mayúsculas los campos de provincia y quitaremos las tildes, para poder cruzar con el dataframe que contiene la dimensión con las provincias normalizadas, los ids de sites y el nombre de los sites.

16. Tras obtener los campos a partir de los cuales podremos obtener los plazos de resolución y restauración de cada ticket, llegamos a la parte principal del proceso, comprobar si cada uno de los tickets cumple estos objetivos propuestos.

    - A partir de la prioridad y el id del site, obtendremos los objetivos de resolución y restauración de cada ticket al cruzar contra la dimensión dim_c001_cumplir_priosite
    - Un ticket habrá cumplido el objetivo de resolución, si su tiempo de primera solución es inferior al tiempo objetivo.
    - Un ticket habrá cumplido el objetivo de restauración, si su tiempo de última solución es inferior al tiempo objetivo.

B.2. Manual de Despliegue

Para poder despleguar el sistema en las condiciones descritas durante el proyecto, se ha alojado en la nube todo el contenido necesario para ello, a excepción de los ficheros de entrada de datos ya que son propiedad de la empresa NTT Data, S.L.U. Todo ello se encuentra disponible en la siguiente dirección:

`https://drive.google.com/drive/folders/1lfWTkkd6QN2-gLXa90KspsZEisNMSgok?usp=sharing`

En este directorio se encuentra el siguiente contenido:

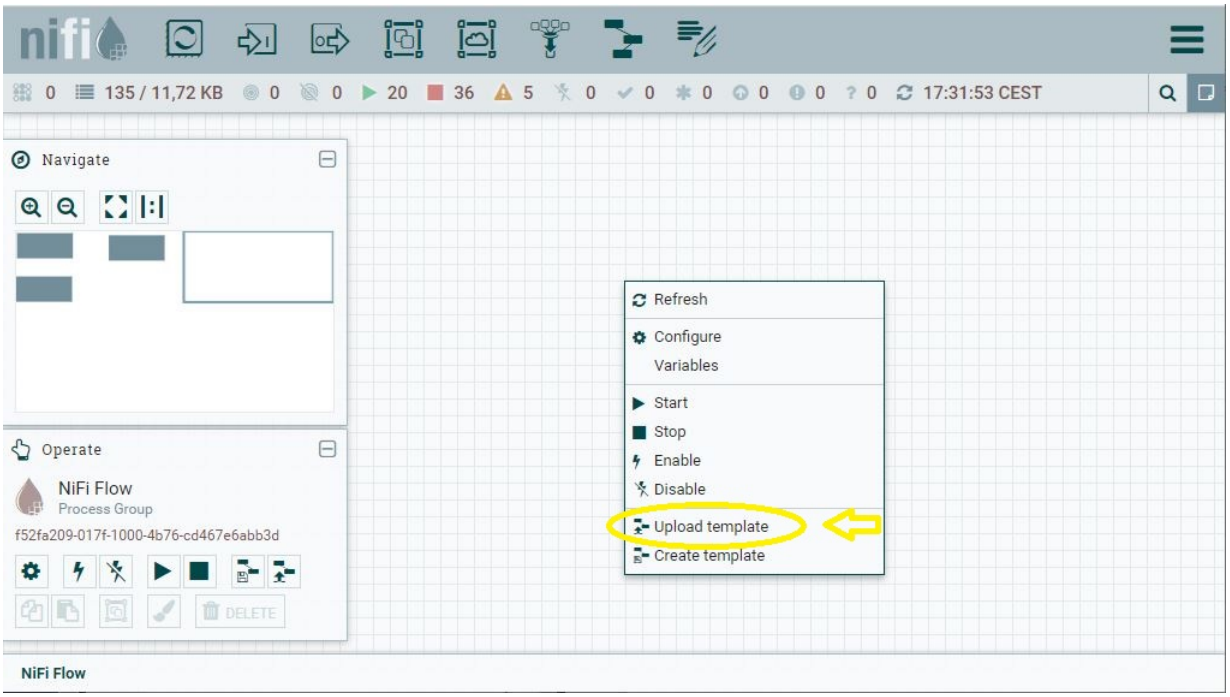
- Un fichero *xml* que contiene el flujo de datos para realizar la ingesta a través de Nifi.
- Un archivo comprimido con la estructura de directorios que se ha utilizado para desarrollar el sistema y cuyo contenido se encuentra descrito en el apartado 5.6. Este archivo debe descomprimirse dentro del directorio raíz 'Disco Local (C:)'
- Un archivo *sql* para la creación (Sin datos) de la BBDD 'bbdd_cumplimientos_sla' y las tablas utilizadas en el proyecto.
- Un archivo *txt* con las versiones de las librerías instaladas, utilizado en el apartado A.2
- Por último se incluye un archivo pdf con la memoria del proyecto.

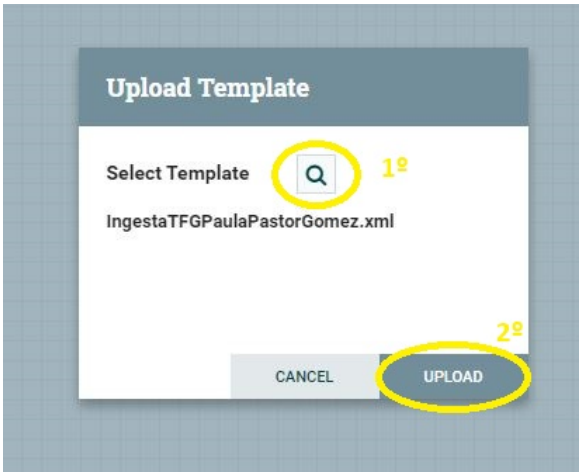
Figura B.1: Contenido Drive

B.2.1. Importar plantilla NIFI

Tras iniciar el proceso nifi en nuestro equipo accedemos a `http://localhost:8080/nifi/` en nuestro navegador web. A continuación se abrirá la interfaz de usuario de nifi, la cual estará vacía si no se ha creado ningún flujo de datos aún.

1. En primer lugar hacemos click derecho sobre el lienzo en blanco y seleccionamos la opción '*Upload template*'. Saldrá una ventana emergente donde podremos buscar el fichero *xml* que queremos importar.



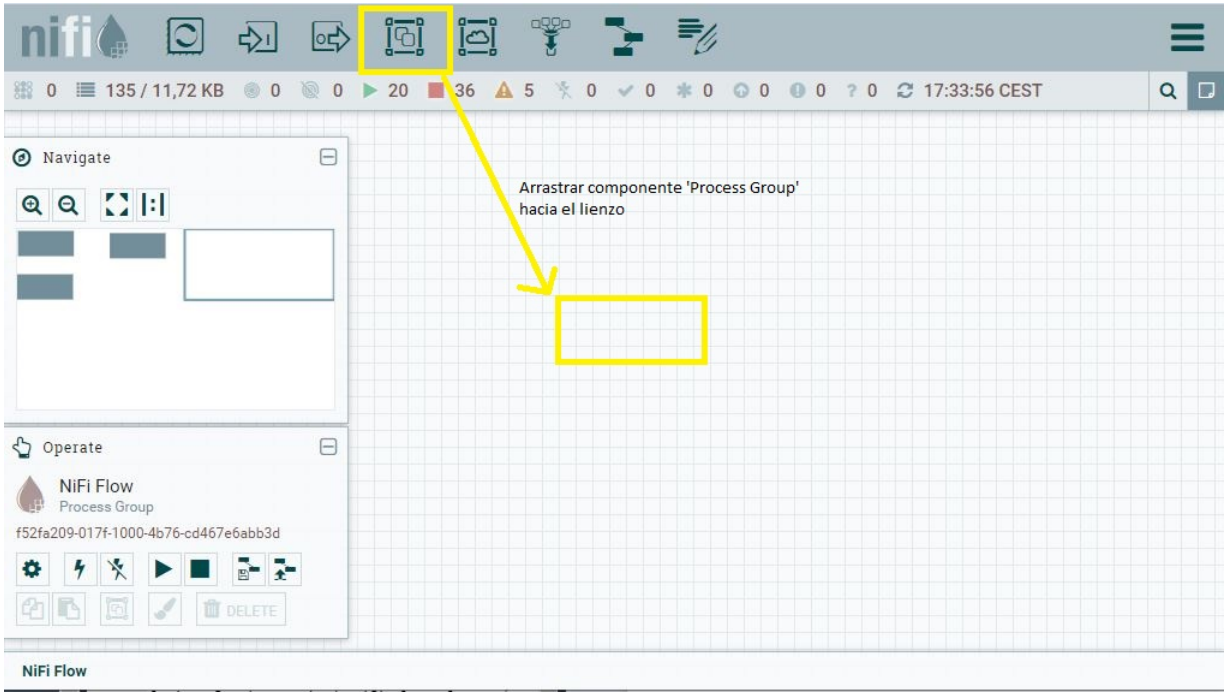


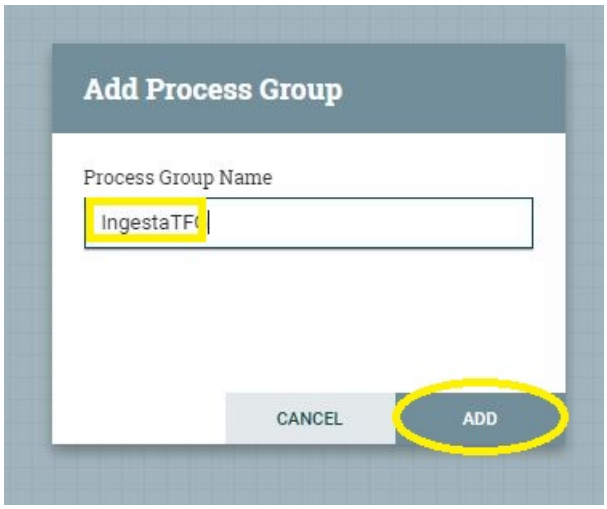
2. Una vez importado, debemos agregar un nuevo componente 'Process Group' a nuestro lienzo, dentro del cual cargaremos el contenido del template. Para ello debemos arrastrar este hacia el lienzo. Aparecerá una ventana donde podremos darle el nombre que queramos.

3. A continuación accedemos a nuestro '*Process Group*' haciendo doble click sobre él.

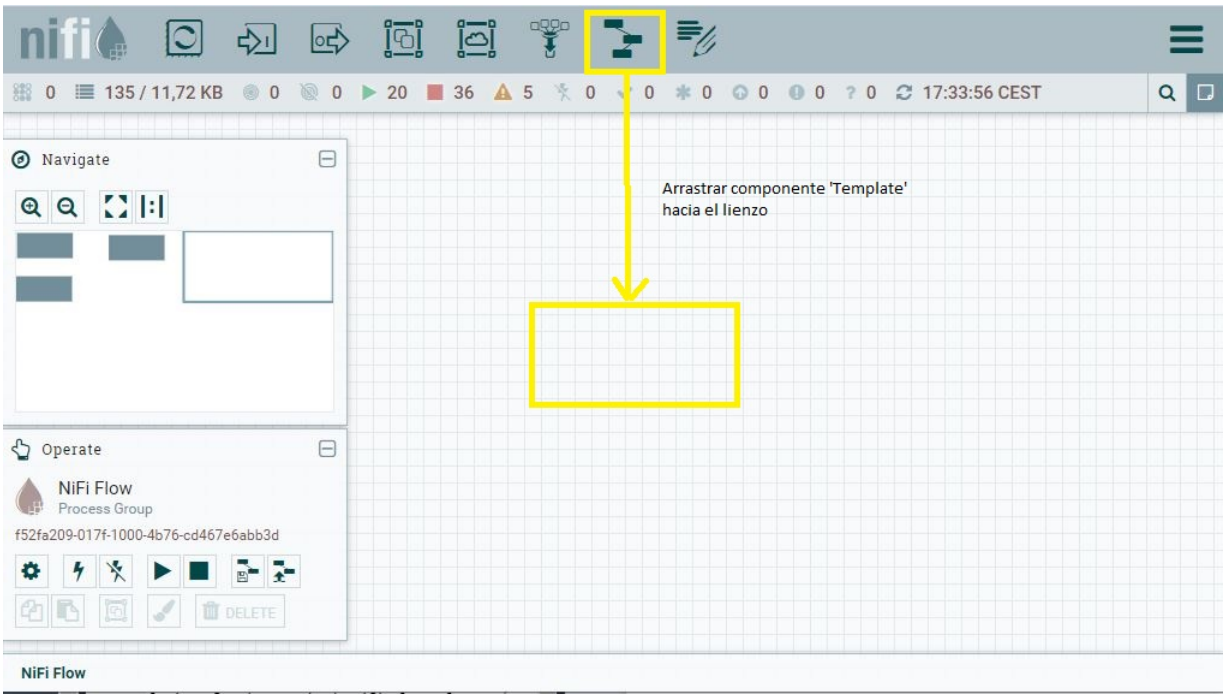
4. Una vez dentro, debemos agregar el contenido del template, arrastrando el componente '*Template*' del menú hacia el lienzo. En la ventana que aparecerá buscamos y añadimos el template.

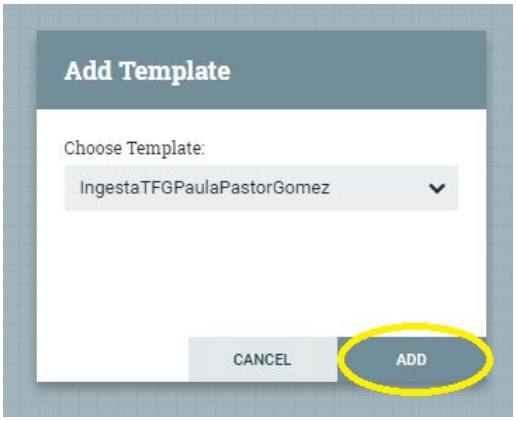
5. Por último iniciamos el flujo de datos. Realizamos doble click sobre el lienzo y seleccionamos la opcion '*Start*'

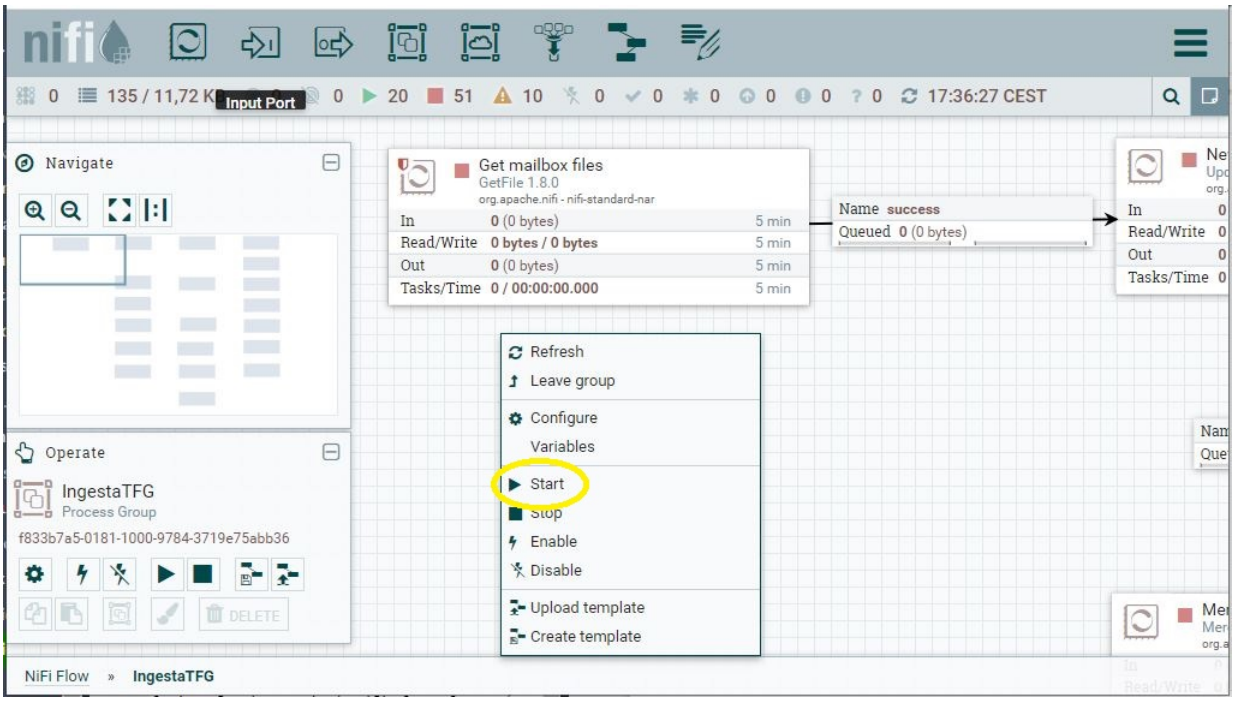

Navigate
Arrastrar componente 'Process Group'
hacia el lienzo
Operate
NiFi Flow
Process Group
f52fa209-017f-1000-4b76-cd467e6abb3d
DELETE
NiFi Flow


Add Process Group
Process Group Name
IngestaTF
CANCEL
ADD


IngestaTFG
Queued
0 (0 bytes)
In
0 (0 bytes) → 0
5 min
Read/Write
0 bytes / 0 bytes
5 min
Out
0 → 0 (0 bytes)
5 min


0 135 / 11,72 KB 0 0 20 36 5 0 0 0 0 0 0 17:33:56 CEST
Navigate
Arrastrar componente 'Template'
hacia el lienzo
Operate
NiFi Flow
Process Group
f52fa209-017f-1000-4b76-cd467e6abb3d
DELETE
NiFi Flow


Add Template
Choose Template:
IngestaTFGPaulaPastorGomez
CANCEL
ADD


0 135 / 11,72 K Input Port 0 20 51 10 0 0 0 0 0 0 17:36:27 CEST
Navigate
Get mailbox files
GetFile 1.8.0
org.apache.nifi - nifi-standard-nar
In 0 (0 bytes) 5 min
Read/Write 0 bytes / 0 bytes 5 min
Out 0 (0 bytes) 5 min
Tasks/Time 0 / 00:00:00.000 5 min
Name success
Queued 0 (0 bytes)
In 0
Read/Write 0
Out 0
Tasks/Time 0
Refresh
Leave group
Configure
Variables
Start
Stop
Enable
Disable
Upload template
Create template
Operate
IngestaTFG
Process Group
f833b7a5-0181-1000-9784-3719e75abb36
DELETE
NiFi Flow » IngestaTFG