UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería Informática

Solución MDM para la gestión de datos maestros de clientes

Trabajo Fin de Grado

Grado en Ingeniería Informática

AUTOR/A: Menzel Gutierrez, Andres Alexander

Tutor/a: Valderas Aranda, Pedro José

CURSO ACADÉMICO: 2021/2022

Dedicatoria

Quiero dedicar este trabajo de fin de grado a todas las personas que me rodean y me han ayudado a crecer como persona.

A mis padres y hermana por el apoyo constante y siempre creer en mi.

A mis amigos por alegrarme los días y llenarlos de felicidad.

A mi perro Barrett por el amor incondicional.

A Sara De La Cruz por estar a mi lado <3 .

Agradecimientos

Me gustaría agradecer a mi tutor Pedro Valderas, gracias a él encontré mi amor en la informática, las ETL, que además me apoyó en la elaboración de este trabajo de fin de carrera.

También me gustaría dar un millón de gracias a mi mentor de Infoverity, Enrique Belda, que me enseñó todo lo necesario para llevar a cabo este proyecto.

Resumen

En la actualidad, los datos almacenados en sistemas informáticos aumentan sin cesar a cada segundo que pasa. Esto incluye datos de diversa índole, yendo desde información personal de particulares hasta información de grandes entidades empresariales y organizativas. Esta enorme cantidad de datos no se almacenan de forma casual, sino que se registran desde un contexto empresarial con el fin de generar más beneficios para las empresas. Para ello, han de ser gestionadas e interpretadas de forma óptima. Esto se puede conseguir, entre otros métodos, mediante el uso de tecnologías vanguardistas que apliquen análisis de datos como el aprendizaje automático (*machine learning*), lo que permite alimentar al sistema con conjuntos de datos de clientes para aprender patrones de compra o comportamiento y sugerir productos específicos a los clientes con más probabilidades de ser comprados. Sin embargo, este proceso se ve entorpecido cuando los datos de la organización provienen de diferentes sistemas de almacenamiento de datos, ERP (*Enterprise Resource Planning*), ya que hay que orquestar los distintos datos que provienen entre las ERP. De no ser así, el proceso no funcionará correctamente, independientemente de lo avanzado que sea el aprendizaje automático disponible u otros tipos de análisis de datos. Se debe buscar una solución siempre que la calidad de la información presentada sea pobre o esté mal estructurada.

En el presente estudio se recrea de forma simplificada cómo una entidad organizativa gestiona la obtención de información procedente de varios sistemas de gestión de datos. El estándar de calidad de información es bajo (lo cual es perjudicial para la empresa) e incluye información repetida en distintos sistemas, registros de clientes redundantes pero con información dispar o registros con información incompleta. Se alcanza la solución mediante el uso de una herramienta ETL (Extracción Transformación Carga), con la que se extraerán los datos de los distintos sistemas del cliente, usando expresiones simples que aplicarán la primera capa de limpieza de datos y estos datos serán cargados al sistema MDM (*Master Data Management*). La herramienta MDM, eje principal de este proyecto, es la encargada de la creación de datos maestros mediante la comparación de los datos procedentes de todos los sistemas. Su puesta en práctica se basa en la aplicación de reglas que combinan datos redundantes, dando lugar a la fusión de los datos y obteniendo tras el proceso unos datos maestros limpios, sin duplicados o registros obsoletos.

Este proceso permite concluir que MDM es una herramienta extremadamente potente para grandes organizaciones que dependen de varios sistemas de datos diferentes. Haciendo uso de ella, pueden obtener un flujo de información carente de errores para comenzar a explotarlo de manera completamente eficiente.

Palabras clave: Extracción transformación carga, Master data management, Enterprise resource planning

Resum

En l'actualitat, les dades emmagatzemades en sistemes informàtics augmenten sense parar a cada segon que passa. Això inclou dades de diversa índole, anant des d'informació personal de particulars fins a informació de grans entitats empresarials i organitzatives. Aquesta enorme quantitat de dades no s'emmagatzemen de manera casual, sinó que es registren des d'un context empresarial amb la finalitat de generar més beneficis per a les empreses. Per a això, han de ser gestionades i interpretades de manera òptima. Això es pot aconseguir, entre altres mètodes, mitjançant l'ús de tecnologies avantguardistes que apliquen anàlisis de dades com l'aprenentatge automàtic (*machine learning*), la qual cosa permet alimentar al sistema amb conjunts de dades de clients per a aprendre patrons de compra o comportament i suggerir productes específics als clients amb més probabilitats de ser comprats. No obstant això, aquest procés es veu entorpit quan les dades de l'organització provenen de diferents sistemes d'emmagatzematge de dades, ERP (*Enterprise Resource Planning*), ja que cal orquestrar les diferents dades que provenen entre les ERP. De no ser així, el procés no funcionarà correctament, independentment de l'avançat que siga l'aprenentatge automàtic disponible o altres tipus d'anàlisis de dades. S'ha de buscar una solució sempre que la qualitat de la informació presentada siga pobra o estiga mal estructurada.

En el present estudi es recrea de forma simplificada com una entitat organitzativa gestiona l'obtenció d'informació procedent de diversos sistemes de gestió de dades. L'estàndard de qualitat d'informació és baix (la qual cosa és perjudicial per a l'empresa) i inclou informació repetida en diferents sistemes, registres de clients redundants però amb informació dispar o registres amb informació incompleta. S'aconsegueix la solució mitjançant l'ús d'una eina ETL (Extract Transform Load), amb la qual s'extrauran les dades dels diferents sistemes del client, usant expressions simples que aplicaran la primera capa de neteja de dades i aquestes dades seran carregades al sistema MDM (Master Data Management). L'eina MDM, eix principal d'aquest projecte, és l'encarregada de la creació de dades mestres mitjançant la comparació de les dades procedents de tots els sistemes. La seua posada en pràctica es basa en l'aplicació de regles que combinen dades redundants, donant lloc a la fusió de les dades i obtenint després del procés unes dades mestres netes, sense duplicats o registres obsolets.

Aquest procés permet concloure que MDM és una eina extremadament potent per a grans organitzacions que depenen de diversos sistemes de dades diferents. Fent ús d'ella, poden obtindre un flux d'informació mancada d'errors per a començar a explotar-lo de manera completament eficient.

Paraules clau: Tranformació de la càrrega de informació, gestió de dades mestres, planificació de recursos empresarials

Abstract

At the present time, the amount of data stored on computer systems is constantly increasing every second. This includes data of various kinds, ranging from personal information of individuals to information of large corporate and organizational entities. This enormous amount of data is not stored accidentally, but is stored because of an entrepreneurial approach in order to generate more profit for companies. To this end, they have to be optimally managed and interpreted. This can be achieved, among other methods, through the use of cutting-edge technologies that apply data analytics such as machine learning, which allows the system to be fed with customer datasets to learn buying or behavioral patterns and suggest specific products that are most likely to be purchased by the customers. However, this process is hindered when the organization's data comes from different ERPs (Enterprise Resource Planning), as the different data coming from different ERPs must be orchestrated. Otherwise, the process will not work properly, regardless of how advanced machine learning or other types of data analytics are available. A solution must be provided whenever the quality of the information presented is insufficient or poorly structured.

This study recreates, in a simplified way, how an organizational entity manages the sourcing of information from various data management systems. Information quality standards are low (which is detrimental to the company) and include repeated information in different systems, redundant customer records but with disparate information or records with incomplete information. The solution is achieved through the use of an ETL (*Extract Transform Load*) tool, with which data will be extracted from the different client systems, using simple expressions that will apply the first layer of data cleansing, and this data will be loaded into the MDM system (*Master Data Management*). The MDM tool, which is the main focus of this project, is responsible for the creation of master data by comparing data from all systems. Its implementation is based on the application of rules that combine redundant data, resulting in the merging of data and obtaining clean master data after the process, without duplicates or obsolete records.

This process leads to the conclusion that MDM is an extremely powerful tool for large organizations that rely on several different data systems. By making use of it, they can obtain an error-free flow of information to start exploiting it in a fully efficient way.

Key words: Extract transform load, Master data management, Enterprise resource planning

Índice general

Índice de figuras

Índice de tablas

CAPÍTULO 1
Introducción

Capítulo donde se introduce al lector el contexto, motivación, objetivos, metodología y tareas principales que componen el presente trabajo, así como una descripción de la estructura general de la memoria.

1.1 Contexto y Motivación

«*La información es una fuente de aprendizaje. Pero a menos que se organice, se procese y se ponga a disposición de las personas adecuadas en un formato para la toma de decisiones, es una carga, no un beneficio*». Esta frase se atribuye al físico estadounidense William Grosvenor Pollard. El trabajo descrito en este documento trata precisamente de eso: organizar, procesar y disponer la información de tal manera que facilite el proceso de la toma de decisiones en el ámbito empresarial.

La Gestión de Datos Maestros, o más conocido por sus siglas en inglés MDM (*Master Data Management*), es un conjunto de herramientas y métodos para gestionar los datos de una empresa relacionados con sus necesidades empresariales, como bien pueden ser las ventas, el marketing y/o las estrategias operativas. La función de un MDM es garantizar la integridad y unicidad de los datos de una empresa con el fin de que todos los servicios puedan acceder a datos precisos, relevantes y actualizados en todo momento desde un único punto de acceso.

Hoy en día, todas las empresas tienen acceso a un gran volumen de conjuntos de datos e información que son dispares y dispersos [1]. Estos son generados tanto por el software interno como por fuentes externas y tienden a agruparse en diferentes formatos y modos de almacenamiento. Las empresas son conscientes de que necesitan controlar, ordenar y evaluar toda esta información para poder explotarla con confianza [2], sobre todo cuando una gestión inadecuada de los datos puede llevar (y lleva, en muchas ocasiones) a perdidas millonarias.

Este trabajo se ajusta a las diferentes líneas de negocio que se abordan en la empresa en la que el autor se encuentra empleado actualmente, Infoverity. Esta empresa se especializa en el tratamiento masivo de datos y este proyecto parte del proceso aprendizaje sobre manejo de datos que el autor ha podido realizar con ellos. Con el presente trabajo se pretende exponer el potencial que tienen las soluciones MDM en este tipo de sector empresarial actualmente.

1.2 Objetivos

Este trabajo consta de dos objetivos principales: El primero es mostrar las posibilidades que ofrece el MDM, específicamente en el ámbito del tratamiento masivo de datos empresariales; y el segundo consiste en afianzar y profundizar los conocimientos sobre tratamiento de datos obtenidos tras cursar el Grado de Ingeniería Informática, siendo este un objetivo personal. Estos dos grandes objetivos pueden dividirse a su vez en hitos, es decir, objetivos más simples que permitan por un lado visualizar el trabajo realizado y por otro establecer y planificar una serie de metas concretas a lo largo del desarrollo del proyecto.

Los hitos relacionados con el objetivo técnico principal son los siguientes:

- Exponer el amplio abanico de posibilidades que ofrece MDM en cuanto a las de estrategias de fusión de datos.
- Demostrar el potencial de usar una herramienta ETL en combinación con el proceso de creación de datos maestros.
- Demostrar lo sencillo que es conseguir datos maestros una vez configurado el sistema de forma correcta, en comparación con la configuración previa.

En cuanto a hitos relacionados con el objetivo personal principal se encuentran los descritos a continuación:

- Exponer un ámbito de la informática no tan conocido como es el de los datos maestros, además de qué son y cómo obtenerlos.
- Aplicar los conocimientos de la asignatura Integración de aplicaciones (IAP), proveniente de la rama de tecnologías de la información.
- Plasmar los conocimientos obtenidos por el autor tras haber trabajado más de quince meses en la empresa Infoverity, especializándose en prácticas MDM.

1.3 Estructura de la Memoria

En esta sección se detalla la estructura de la memoria, con el objetivo de que el lector pueda orientarse de forma sencilla por la misma. Los capítulos que la componen se resumen a continuación (a excepción del capítulo 1):

- **Capítulo 2, Master Data Management:** Este capítulo expone los conocimientos necesarios para entender el MDM en su contexto, junto con por qué es necesario y cuáles son sus características esenciales.
- **Capítulo 3, Contexto tecnológico para las soluciones MDM:** Se comentan las diferentes tecnologías que se han empleado para realizar el proceso de MDM, así como herramientas alternativas que se plantean en un principio y la comparación entre dichas alternativas y las herramientas finalmente utilizadas.
- **Capítulo 4, Caso de estudio:** Se propone un caso de estudio de ejemplo sobre el que aplicar las herramientas MDM y obtener un resultado del que extraer conclusiones. Pese a ser un caso modelo simulado con el mero fin de ejemplificar, su estructura parte de la de casos reales en empresa.

- **Capítulo 5, Solución desarrollada:** Este capítulo explica con detalle la arquitectura de la solución, así como cuáles han sido las aplicaciones utilizadas, cómo y por qué se han utilizado, cuál es su objetivo y cómo interactúan entre ellas. Además, se propone la solución MDM escogida.
- **Capítulo 6, Resultados experimentales y discusión:** Se llevan a cabo una serie de ejecuciones sobre el modelo para verificar las hipótesis teóricas previamente expuestas y se analizan los resultados obtenidos con el fin de determinar la validez de la propuesta.
- **Capítulo 7, Conclusiones y Trabajo Futuro:** Para cerrar el proyecto, de acuerdo con el análisis expuesto en el capítulo anterior, se exponen una serie de conclusiones extraídas del trabajo previo. También se concreta la línea de trabajo que el proyecto adoptará si continúa en el futuro junto con posibles ampliaciones del proyecto.
- **Anexo, Objetivos de desarrollo sostenible:** Se sitúa el proyecto dentro de los objetivos de desarrollo sostenible comunes a todos los Trabajos de Fin de Grado.
- **Apéndice A, Manual de instalación:** Se describe de forma precisa el proceso de instalación de las herramientas que se han usado para llevar acabo este proyecto.

CAPÍTULO 2
Master Data Management

En este capítulo se define MDM, se explica por qué es necesario y se comentan ciertas características que presentan habitualmente los procesos de tratamiento de datos mediante MDM hoy en día.

2.1 ¿Qué es el Master Data Management?

El MDM es una disciplina que se basa en gran medida en los principios de gobernanza de datos con el objetivo de crear una visión confiable y autorizada de los datos de una empresa, intentando unificarlos en la medida de lo posible. Las organizaciones dan cada vez más importancia a las decisiones basadas en los datos en el mercado global actual y, a medida que un número cada vez mayor de sistemas aportan registros digitales de las personas, los lugares y las cosas que más importan a una empresa.

Como tecnología, las soluciones MDM automatizan el modo en que los datos críticos para el negocio se gobiernan, gestionan y comparten a través de las aplicaciones utilizadas por las líneas de negocio, las marcas, los departamentos y las organizaciones. MDM aplica la integración, la reconciliación, el enriquecimiento, la calidad y la gobernanza de los datos para crear registros maestros. La automatización se utiliza para identificar, cotejar y fusionar datos en los sistemas que los contienen, y luego los datos limpios se comparten con las aplicaciones, los sistemas y los análisis que los necesitan. Al fusionar los registros, MDM también puede corregir las incoherencias en los registros, capturar la procedencia de los datos y crear una pista de auditoría de los cambios. Esto proporciona transparencia para simplificar cómo se crea o modifica cada registro maestro.

En conclusión, el MDM es una disciplina que permite generar unos datos maestros confiables y utilizarlos para orientar la toma de decisiones de un negocio y facilitar sus procesos productivos.

2.1.1. Tipos de datos en una empresa

Si bien el objetivo final es convertir todo nuestro volumen de datos en datos maestros, cómo ya se ha comentado, existen otros tipos de datos. A continuación se procede a definirlos brevemente y comentar sus características. Existen los siguientes tipos:

- **Datos transaccionales:** Uno de los tipos de datos más importantes para muchas organizaciones son los datos transaccionales. Pueden referirse a facturas, entregas, ventas, devoluciones, tiques de problemas, reclamaciones u otros documentos si-

milares. Cada vez que se produce una transacción, se generan datos que pueden ser rastreados.

- **Metadatos:** Muchos sistemas informáticos y otras soluciones generan metadatos, como documentos XML, descripciones de columnas de bases de datos, archivos de registro y similares.
- **Datos de referencia:** Estos datos suelen utilizarse para relacionar y utilizar datos e información que se extienden más allá de la propia organización. Por ejemplo, una empresa puede controlar los tipos de cambio o las políticas gubernamentales, entre otros factores.
- **Datos jerárquicos:** Estos datos muestran y rastrean la relación entre los datos. Los datos jerárquicos pueden formar un sistema de contabilidad y también pueden proporcionar descripciones de las relaciones del mundo real, como el organigrama de una empresa. Algunos consideran los datos jerárquicos como un súper dominio MDM, ya que son fundamentales tanto para descubrir como para comprender la relación entre los diferentes datos.
- **Datos no estructurados:** Los datos no estructurados no se adhieren a un modelo predefinido o no están organizados de manera predefinida. A menudo, estos datos contienen mucho texto y pueden incluir documentos, correos electrónicos, archivos PDF, etc.

  Para ilustrar la jerarquía que existe en la calidad de los datos se ha elaborado la figura (2.1). En ella se muestra cómo los datos maestros son los más valioso y los que menos esfuerzo requieren para ser interpretados. El resto de tipos de datos están ordenados conforme a su dificultad de interpretación, tratamiento y conversión a datos maestros.

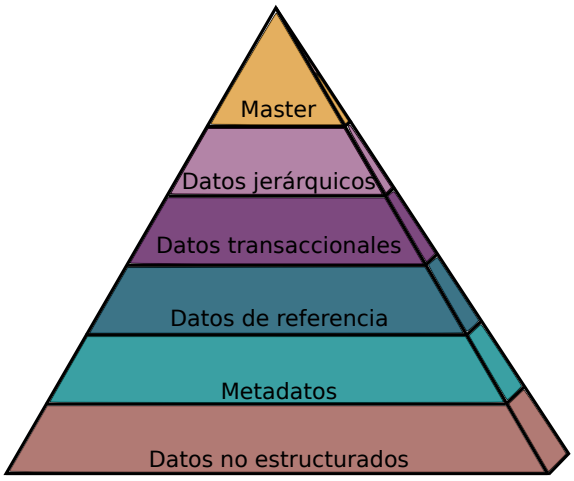

Figura 2.1: Jerarquía de calidad de los datos.

El objetivo del MDM es convertir toda esta información en datos maestros, de manera que pase de ser información no agrupada a información interpretable. El porqué es necesario tener unos datos maestros de calidad y sus consiguientes beneficios se detallan en la siguiente sección de este trabajo.

2.2 ¿Por qué es necesario el MDM?

La creciente demanda de herramientas de gestión de datos maestros de primera calidad se puede atribuir a la necesidad de superar los retos de la gestión manual de datos,

los datos redundantes y/o no actualizados y las discrepancias de datos tanto en los datos maestros de una organización como en los datos empresariales. También proporciona una visión más profunda de la calidad de la interacción de su empresa con sus clientes. A continuación se enumeran algunos de los principales beneficios que se observado en la utilización de una herramienta MDM:

- La gestión de datos maestros permite la gestión centralizada y descentralizada de los datos. Esto significa que puede gestionar los datos maestros asignándolos a un único usuario o descentralizarlos adjudicando su propiedad a múltiples entidades.
- Una solución de Gestión de Datos Maestros puede configurarse rápidamente, sin necesidad de ninguna codificación, lo que la convierte en la opción más veloz y fácil de usar que existe.
- Los usuarios clave de la empresa asignados a la gestión de los datos maestros pueden hacer uso de ello sin esfuerzo adicional ni tener que esperar a la ayuda de su departamento tecnológico.
- Los administradores y/o gestores de datos pueden recibir la autoridad para aprobar o rechazar los datos desde una bandeja de entrada central, asegurando así que el control de calidad de los datos se centraliza y se contabiliza.
- Una vez organizados los datos no será necesario reorganizarlos ya que, al estar estructurados como datos maestros, los nuevos datos entrantes se colocarán en las nuevas estructuras de datos generadas.

Esto no quiere decir que la implementación de MDM esté completamente libre de desafíos. Pero, como se ha comprobado durante la realización del trabajo a través de la experiencia personal profesional con administradores de datos y directores de negocio de varias empresas, es algo imprescindible para cualquier empresa que maneje grandes volúmenes de datos. Al invertir en una solución de gestión de datos maestros, la calidad de los datos mejora, así como las operaciones empresariales. Al confiar en una única fuente de verdad en todas las entidades de la empresa y en múltiples sistemas, se garantiza la integridad de los datos de esta empresa. Y lo que es más importante, permite ser completamente transparente y rendir cuentas en todo momento, lo que proporciona el activo más valioso de su organización: la confianza de los clientes.

Un buen ejemplo de esto serían los problemas que aparecen cuando se parte de varias bases de datos con transacciones (de compra) de los clientes. Tener que cotejar, comprobar y comparar los datos no solo ralentiza su análisis sino que además da pie a fallos en la fiabilidad de los datos. Mediante herramientas MDM es posible acelerar el análisis y generar agrupaciones de datos confiables.

2.3 Aspectos a tener en cuenta en un análisis orientado a MDM

A través de la experiencia en empresa y la constante evolución de las técnicas de MDM se pueden definir una serie de directrices o buenas prácticas que, por lo general, son convenientes aplicar cuando se trabaja con datos para generar un conjunto de datos maestros. Aunque podrían extraerse más, a continuación se detallan las directrices que son esenciales:

- **Recoger toda la información posible:** Cuanta más información se posea sobre los clientes de la empresa, mejor. Esto hace que dicho negocio sea más competitivo. Analizar cómo llegan los datos a esta empresa ayudará a mejorar el proceso y a decidir qué nuevas estrategias puede la empresa aplicar en este sentido.
- **Comprobar la calidad de la información:** No se debe limitar el proceso a sumar más y más datos. La información que recojamos debe ser de alto nivel. Para ello, la empresa debería auditar los datos de forma periódica y aleatoria.
- **Crear una capa de metadatos común:** Otra de las mejores prácticas de gestión de datos maestros es la organización de una capa de metadatos, que permita compartir la información desde cualquier punto del proceso de administración y análisis. De este modo, se puede agilizar y optimizar el uso de los datos.
- **Organizar los datos:** Si toda la información recopilada se almacena en una estructura organizada de archivos de datos, se puede recuperar de la forma más sencilla y práctica posible. Esto reduce el tiempo de búsqueda y agiliza el ritmo de trabajo.
- **Mejorar el acceso a los datos:** Eliminar todas las barreras posibles entre los datos y las personas que los van a utilizar es otra de las mejores prácticas de la gestión de datos maestros. Nos permite mejorar la productividad y evitar confusiones que puedan interferir en el desarrollo del negocio.
- **Mejorar la ciberseguridad:** Tanto los archivos internos como los adquiridos en el mercado deben estar protegidos para evitar que la información útil para el negocio se vea comprometida por un ciberataque.
- **Concienciar a los empleados:** Todas las personas implicadas en la gestión de los datos de la empresa, independientemente de su función, deben ser conscientes de su responsabilidad en la protección de la información. De este modo se reduce la posibilidad de que se produzcan errores humanos que puedan afectar a su integridad.
- **Proporcionar la formación adecuada:** Toda empresa debe proporcionar la mejor formación posible a sus empleados. Los empleados a su vez deben ser capaces de entender lo que significa la gestión de la información y saber cómo utilizar las herramientas que tienen a su disposición.
- **Planificar la gestión de la información:** Las mejores prácticas de gestión de datos maestros van más allá de la protección adecuada de los datos. También es primordial planificar todos los pasos que se darán en el proceso de gestión de la información. Tener un plan a largo plazo permite conocer el devenir de los datos están siendo manejados.
- **Búsqueda de innovaciones tecnológicas:** Las nuevas tecnologías evolucionan rápidamente y periódicamente ofrecen nuevas soluciones para facilitar la gestión de los datos de la empresa. Estar al tanto de los últimos avances en este campo permite mejorar la gestión de esta información.

2.4 MDM y la gestión de la seguridad de la información

Las organizaciones corren constantemente el riesgo de recibir (y pagar) una fuerte sanción si no cumplen con las normas y directrices sobre seguridad de la información y transmisión de los datos. El cumplimiento abarca más allá de la normativa de protección

de datos, contemplando otras normativas globales y regionales, mandatos específicos del sector y contratos específicos de los socios comerciales. Esta sección se establece entorno a la normativa europea de protección de datos, General Data Protection Regulation (GDPR) [3].

2.4.1. MDM y la GDPR

El GDPR tiene como objetivo fundamental proteger la privacidad de los datos de las personas dentro de la Unión Europea (UE) y el Espacio Económico Europeo (EEE), dando a los individuos un mayor control sobre sus datos personales y sobre cómo se recogen y utilizan. Esto significa que las organizaciones deben tener un control estricto de los sistemas y procesos que utilizan para recoger, gestionar y compartir información sobre sus clientes, empleados, proveedores y otras partes con las que hacen negocios.

Sin embargo, la realidad es que los datos privados sobre las personas están fragmentados en múltiples silos departamentales y cada departamento recoge y utiliza los datos a su manera. Esto lleva a algunas organizaciones a no identificar si los datos privados que residen en dos aplicaciones departamentales distintas se refieren a la misma persona, lo que pone en grave peligro el cumplimiento del GDPR. Una solución de MDM aborda este problema proporcionando una aplicación centralizada en la que los datos maestros relacionados con un individuo pueden ser centralizados, limpiados y con referencias cruzadas para crear una representación coherente del individuo en toda la empresa.

2.4.2. Derecho de acceso y borrado

El GDPR establece que los sujetos de los datos tienen derecho a solicitar y obtener sus datos recogidos por las organizaciones y, en ciertas condiciones, también a que sus datos sean borrados. Una vez más, cuando se trata de datos maestros, no hay mejor lugar para orquestar un borrado completo (o acceso) que los datos maestros del individuo, ya que están centralizados. Del mismo modo, una solicitud de borrado puede ser orquestada a través de un flujo de trabajo guiado en el que los datos maestros se borran en la propia aplicación MDM, así como en todas las demás aplicaciones empresariales en las que se comparten los datos maestros.

2.5 Mejora del rendimiento del negocio

Tras ver las posibilidades del MDM aplicado a negocios y cómo puede mejorar sus estructuras productivas, se procede a concretar su uso, utilizando la filial española de Puma para ejemplificar este fenómeno. A continuación se exponen sus necesidades, el reto administrativo que conlleva esta filial y cómo mediante MDM se resuelven estos problemas[4].

2.5.1. Necesidades de la empresa

Puma, siendo una empresa internacional, necesitaba publicar de manera eficiente los productos tanto para Europa como para mercados internacionales. Además necesitaba mantener las jerarquías de comercio electrónico dentro de una plataforma de gestión centralizada.

Otra de sus necesidades era tener la capacidad para crear y definir los atributos concretos de cada producto para los mercados japoneses y asiáticos, tales como el tallaje o

las descripciones de los productos. Para ello necesitaban agrupar los países de Asia y el Pacífico de manera que sus datos fueran tratables conjuntamente.

Esto generó dos retos principalmente. El primero era la consolidación de los actuales atributos de los productos e identificación de nuevos campos para constituir una introducción de datos estandarizada en un *marketplace* europeo. El segundo era adaptarse al tamaño de gráficos que se utiliza en Asia, que incluyen un conjunto de medidas exactas que deben ser definidas y mantenidas por estilo.

2.5.2. Solución y resultados

Mediante herramientas de MDM se consiguieron las siguientes soluciones:

- Mejoras en la sindicación de productos de Puma utilizando un único formato para múltiples *marketplaces* en Europa.
- Implementación de una nueva jerarquía en la información de gestión de los productos acorde a las necesidades de *eCommerce*, posibilitando herencia automática de atributos en base a grupos dentro de la jerarquía.
- Creación de una nueva capacidad de publicación de productos específica para el mercado japonés, incluyendo el tamaño de gráficos específico para Asia.

CAPÍTULO 3
Contexto tecnológico para las soluciones MDM

En este capítulo se comentan las diferentes tecnologías que se han empleado para realizar el proceso de MDM, así como las demás opciones que se han barajado previamente, y se realiza una comparación entre todas las alternativas y la solución escogida finalmente.

3.1 Tecnologías

Este apartado es un repaso por las tecnologías que han sido contempladas para desarrollar el MDM y su lugar en el contexto tecnológico actual en lo que se refiere a este tipo de soluciones. Se centra en Informatica, empresa desarrolladora de la herramienta MDM (sistema con el que se realiza el desarrollo del proyecto), y sus necesidades. También se comentan posibles alternativas a este método barajadas en un primer momento, aunque sin realizar una descripción tan exhaustiva como la de Informatica. Es relevante conocer las necesidades y capacidades de otras opciones para entender el motivo de la elección final de Informatica.

En el caso concreto de Informatica se emplean dos tecnologías externas, una que funciona como la base de datos y otra que permite el despliegue de aplicaciones basadas en servidor.

Para las bases de datos, las tecnologías contempladas por la documentación de Informatica son:

- **Microsoft SQL Server**
- **IBM Db2**
- **Oracle Database**

Para las plataformas de aplicaciones de servidor, se considera en la documentación las siguientes tecnologías:

- **Red Hat JBoss**
- **WebSphere**
- **Oracle WebLogic**

Se consideran estas tecnologías para ser comentadas por resultar relevantes, ya que Informatica funciona desplegándose sobre estos dos tipos de sistemas. Sin embargo, no

por ello se perderá la oportunidad de comentar en este trabajo las tecnologías alternativas a Informatica, pues resultan de suma importancia para contemplar las posibilidades a la hora de un desarrollo de MDM en la actualidad. Las tecnologías alternativas que se expondrán son Boomi y Stibo por ser las consideradas como más relevantes en la actualidad para el desarrollo de este mismo tipo de actividades.

3.1.1. Informatica

Informatica es una plataforma de gestión de datos inteligente, con administración en la nube y un gran abanico de posibilidades para plantear y desarrollar nuestro almacenamiento y tratamiento de datos de forma eficiente, efectiva, rápida y versátil. Con un entorno listo para *cloud*, pensado para el uso de inteligencia artificial y preparado para la integración de funcionalidades a bajo nivel, Informatica va a facilitar en gran medida el desarrollo del MDM que se utilice de forma específica, posibilitando un despliegue e integración del sistema sencillo y listo para funcionar lo antes posible. Esto sucede incluso si algunas características de su diseño no van a ser utilizadas concretamente en este proyecto, como el aspecto *cloud*.

Las facilidades que aporta Informatica sobre el diseño de MDM es lo que hace que sea la elegida para su utilización en este tipo de proyecto de gestión de datos. Se encuentra en un punto de complejidad de integración, desarrollo y diseño perfecto para los objetivos de estos proyectos y aumenta en gran grado la transparencia del sistema creado durante el desarrollo de los proyectos.

Al ser Informatica la principal tecnología empleada a lo largo del proyecto, permitirá establecer nuestro sistema de MDM multidominio de forma rápida y sencilla. Así, simplificará los procesos a realizar y permitirá integraciones nativas con tecnologías comentadas en el resto del capitulo. De esta forma, se pueden desplegar todos los componentes necesarios para Informatica (incluyendo su integración) de forma organizada y tipificada como documentación oficial del sistema.

3.1.2. Microsoft SQL Server

Microsoft SQL Server es un sistema de gestión de bases de datos relacionales desarrollado por Microsoft. Al ser un servidor de bases de datos, es un producto de software cuya función principal es almacenar y recuperar datos a petición de otras aplicaciones de software, ya sean ejecutadas en el propio ordenador desde el que se va a acceder o en otro ordenador a través de la red. Microsoft comercializa al menos una docena de ediciones diferentes de Microsoft SQL Server, cada una dirigida a un público y necesidad diferentes. Se ocupan de distintas cargas de trabajo que van desde pequeñas aplicaciones de un solo equipo hasta grandes aplicaciones orientadas a Internet con muchos usuarios simultáneos. Se trata de uno de los sistemas de bases de datos más prevalentes y extendidos del mercado, haciéndolo una elección ideal para este trabajo.

Se empleará esta tecnología en el proyecto para el almacenamiento de los datos que posteriormente serán procesados, ya que aporta una estructura perfecta para el trabajo a realizar y es compatible con Informatica, de forma que el cliente puede acceder y funcionar sobre las bases que establece Microsoft SQL Server, lo cual es una gran ventaja.

3.1.3. Red Hat JBoss

Red Hat JBoss Platform es un servicio de creación, despliegue y alojamiento de aplicaciones que ofrece seguridad, rendimiento y escalabilidad de nivel empresarial en cual-

quier entorno. Ya sea de forma local, virtual o en nubes tanto privadas como públicas o incluso híbridas, JBoss EAP puede ayudar a ofrecer aplicaciones de forma más veloz en cualquier lugar.

Para este proyecto también se utiliza Red Hat JBoss como el sistema de alojamiento en el lado de servidor porque, además de ser compatible con Informatica, ofrece grandes capacidades en cuanto a integración y despliegue de los procesos a desarrollar. Además, en la documentación de Informatica existe una guía de implementación del servidor con esta tecnología combinada precisamente con Microsoft SQL Server, visto anteriormente.

3.1.4. IBM Db2

IBM Db2 es un software de manejo de datos diseñado por los principales expertos a nivel mundial en materia de bases de datos. Permite a los desarrolladores, DBAs y arquitectos de datos ejecutar transacciones con baja latencia y analítica en tiempo real equipados para las cargas de trabajo más exigentes. Utilizado por sectores como la banca global o la energía sostenible, Db2 es la base de datos híbrida, resistente y probada en la industria que proporciona la disponibilidad, seguridad y escalabilidad para los sistemas que dirigen el mundo.

Se trata de un sistema probado en su disponibilidad, evitando y facilitando la gestión de interrupciones. Está diseñado para la seguridad, siendo construido con este valor como bandera desde el inicio. Cuenta con un extensivo y complejo sistema de encriptados para proteger los datos del usuario. Contempla la escalabilidad también al nivel más bajo puesto que está pensando para ser capaz de adaptarse a las dimensiones de cualquier solución de forma dinámica y eficiente.

Pese a este gran abanico de características y funcionalidades, su compatibilidad con el sistema de este proyecto no es tan amplia como la que proporciona Microsoft SQL Server, además de que su coste y complejidad de uso y despliegue resultan perjudiciales para el desarrollo del trabajo. Por estas razones, incluso tras valorar las grandes capacidades de Db2, se ha preferido emplear Microsoft SQL Server en su lugar.

3.1.5. WebSphere

La aplicación de servidores IBM WebSphere acelera la entrega de aplicaciones con un entorno de ejecución altamente fiable basado en Java Enterprise Edition. Las empresas ágiles de hoy en día dependen de que los líderes de tecnologías entreguen aplicaciones de alto rendimiento para dar soporte a los cambios rápidos, a la vez que ahorran costes y mejoran el tiempo de obtención de valor. Sin embrago, conseguir estos resultados en entornos tecnológicos complejos y heterogéneos puede ser un reto. WebSphere permite a sus equipos de desarrollo ofrecer nuevas aplicaciones nativas de la nube y modernizar las aplicaciones existentes.

Pese a encontrar este servicio para servidores en un gran abanico de soluciones tecnológicas *online*, su sistema esta orientado firmemente en los sistemas en *cloud*, lo cual no está en la línea de necesidades de este proyecto. Por esta razón, se opta por usar el entorno de Red Hat JBoss, que está más preparado para el trabajo local, como es el caso.

3.1.6. Oracle WebLogic

El servicio de servidor de Oracle WebLogic es una plataforma unificada y extensible para desarrollar, desplegar y ejecutar aplicaciones empresariales como por ejemplo Ja-

va para las instalaciones y la nube. Oracle WebLogic Server ofrece una implementación sólida, basada en la experiencia y escalable de Java Enterprise Edition y Jakarta EE.

Posee una gran cantidad de características importantes para un servicio de servidor, como puede ser la gestión de herramientas y API para la automatización de operaciones, escalado y reinicios automáticos para nodos con interrupciones de modo que se pueda maximizar la disponibilidad, así como despliegues de tecnología Kubernetes para el uso de herramientas de administración.

Como en los casos vistos anteriormente, el problema con este sistema es que su complejidad y cantidad de sistemas y características resultarían más perjudiciales que beneficiosas en el desarrollo de este proyecto, ya que la gestión sobre el sistema requerirá de un mayor grado de especificidad y complejidad que no aportaría nada a la solución que se propone. Por esto, pese a ser compatible, no se hará uso de este sistema aplicado con Informatica.

3.1.7. Boomi

Boomi es una plataforma de integración que conecta de forma inteligente aplicaciones y automatiza los flujos de trabajo, facilitando las tareas de MDM y dotando de transparencia, cohesión y legibilidad al proceso. Conecta todo en su ecosistema digital, incluso en el caso de las instalaciones, con el fin de minimizar las interrupciones, reducir el riesgo y simplemente poder desarrollar estas tareas de forma ágil. Así, facilita enormemente los retos de integración que surgen durante el desarrollo de nuevos modelos de negocio y productos digitales.

Permite enfrentar directamente los datos a través de su interfaz de usuario visual de *arrastrar y soltar* junto con su plataforma de bajo nivel, aumentando la eficiencia y reduciendo los errores con capacidades únicas como el mapeo y la configuración de integración inteligente, así como la prueba y resolución de errores. Además, reúne los datos de los sistemas, las aplicaciones y las personas para facilitar la comprensión por parte de los consumidores y ofrecerles una buena experiencia. Todo esto ocurre desde un entorno *cloud*, en navegador, que permite el acceso y gestión de datos desde cualquier sistema y lugar.

Aunque cuente con todas estas increíbles características, este sistema también cuenta con grandes desventaja. No permite la suficiente personalización para poder ajustarlo a las necesidades del proyecto. Esto podría considerarse compensado con su gran rapidez y eficiencia de ejecución si no fuese porque esta característica no resulta tan crítica para el desarrollo del proyecto en cuestión, como sí lo es la capacidad de ofrecer una solución ajustada a las necesidades establecidas.

3.1.8. Stibo

Stibo se trata de una plataforma para MDM prevalente en el mercado que permite la gestión de los datos de tal forma que se pueda dotar de cohesión y se facilite la condensación de estos. La conexión de los datos se realiza con el fin de aportar una mayor transparencia, legibilidad e interpretabilidad a estos. Permite a su vez la gestión de los datos para garantizar su calidad, eficiencia y agilidad durante su uso, así como habilitar la sincronización y propagación de estos datos maestros en múltiples dominios y ecosistemas. De esta forma se facilita la colaboración.

Cuenta con un objetivo y capacidades similares a las que provistas por Informatica. Sin embargo, un punto a favor para Stibo es que permite un mayor grado de personalización a través de sus funcionalidades. Por ejemplo, es capaz de implementar código

JavaScript en el desarrollo de las operaciones de *Data Governance*, permitiendo un mayor grado de especificidad y precisión en la recolecta de estos.

Por el contrario, también existen problemas en Stibo que hacen de Informatica una solución mas ajustada a las necesidades del proyecto actual, ya que estas mismas funcionalidades que permiten mayor customización también aumentan la complejidad durante la gestión. Esto se debe a que la existencia de grandes restricciones en la obtención de los datos puede entorpecer el procesamiento de los mismos e incluso puede derivar en problemas de rendimiento, en función de la dimensión de las restricciones propuestas. Aunque estas capacidades nos permitan un mayor grado de especificidad y enfoque en nuestro proyecto, se ha valorado que la complejidad añadida generaría más problemas que soluciones, conllevando un peor resultado final, y los objetivos propuestos ya son perfectamente abarcables por Informatica sin riesgos adicionales.

3.2 Conclusión

Dados los resultados del análisis del contexto MDM, se opta por escoger Informatica como sistema principal. Como se ha expuesto, se encuentra en un buen punto medio entre la personalización y especificación de proyectos y la facilidad para su uso, modificaciones y expansiones sobre la solución. Como se ha comentado, Informatica requiere de dos tecnologías más para ser empleada, una que permita la creación y gestión de bases de datos y otra que sea una plataforma de aplicaciones que permita su despliegue y escalabilidad.

A partir del análisis previamente realizado junto con los conocimientos del autor previos en experiencia de uso, se escoge *Windows* como sistema operativo, Microsoft SQL Server como gestor de la bases de datos y Red Hat JBoss como plataforma de aplicaciones web. Estas tecnologías se encuentran implementadas en otras áreas de la empresa, por lo que el equipo está familiarizado con el uso de tecnologías al haber tenido experiencia previa. A su vez, Microsoft SQL Server es un sistema que cumple con las necesidades del proyecto y que no aporta de una complejidad que no se vaya a aprovechar durante el desarrollo de la solución, lo que complicaría el proyecto. Red Hat JBoss se encuentra también en un término medio en cuanto a estos aspectos que se tienen en cuenta, lo que hace que ambas tecnologías sean perfectas para su implementación y para cumplir con las necesidades del proyecto.

Otra ventaja de escoger Informatica como herramienta de desarrollo es que, a nivel empresarial, es la herramienta con más mercado. Dentro de la empresa Infoverity, el 63 % de los ingresos en los 6 primeros meses de 2022 son generados con el *software* de Informatica (3.1) (Información obtenida por la mesa de operaciones de Infoverity).

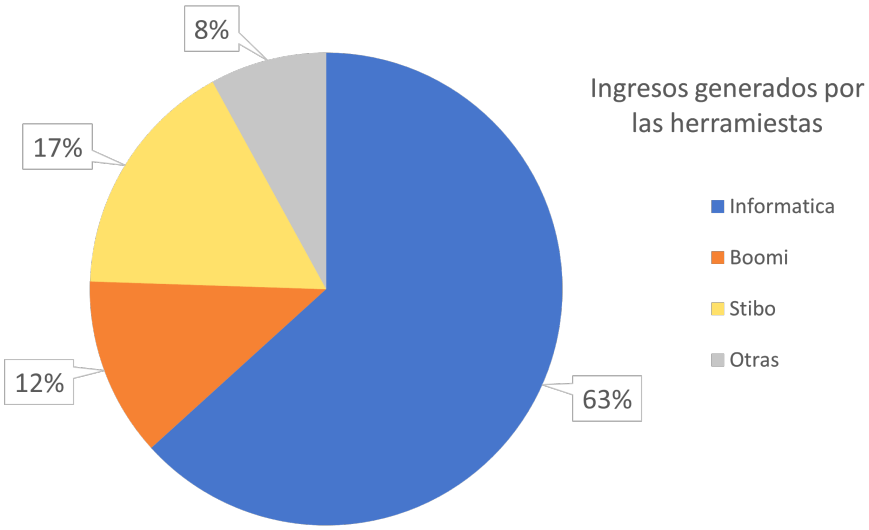

Figura 3.1: Gráfico circular con el porcentaje de ingreso monetario atribuido a herramientas.

CAPÍTULO 4
Caso de estudio

Una vez establecido un contexto tecnológico que da pie a solucionar problemas de gestión de datos, se propone un caso de estudio donde la herramienta MDM gestionará y creará datos maestros. Este capítulo relata con detalle el caso de un negocio con problemas de gestión de datos, lo que ofrece la ocasión ideal para ejemplificar una solución con dicha herramienta.

4.1 Modelo de negocio

El negocio que se presenta es una empresa dedicada a la venta de automóviles que sigue el modelo de comercio detallado a continuación.

Concesionarios

La empresa tiene varios concesionarios afiliados, los cuales tienen entre uno y una gran cantidad de automóviles. Los coches son vendidos a clientes particulares mediante el concesionario.

Clientes

Un cliente puede adquirir uno, varios automóviles o ninguno, pero no tienen porque provenir del mismo concesionario. La empresa también guarda datos personales de clientes que han realizado el proceso de compra tales como dirección, correo electrónico, teléfono y fecha de nacimiento, entre otros valores.

4.2 Modelo de datos

La empresa guarda sus datos en tres sistemas distintos. Esto ocurre porque cada concesionario usa diferentes bases de datos o sistemas para guardar la información. En concreto hay 3 sistemas de bases de datos diferentes, que son Oracle DB, Mongo DB y SAP. En proyectos reales se puede trabajar hasta con once tipos diferentes de bases de datos o sistemas de gestión de datos, entre los cuales la cantidad y calidad de datos varía.

El negocio se da cuenta que entre los sistemas hay información de clientes repetida, es decir, un cliente que ha comprado en dos o más concesionarios. Ese no es el único problema de repetición sino que además en algunos sistemas la información de un cliente está más completa que en otros sistemas, debido a que cuando se formalizó la inscripción del cliente no se guardó toda la información del cliente. Incluso aparecen casos en los que

la combinación de varios registros procedentes de los distintos sistemas mejoraría la calidad de datos de un único registro centralizado, ya que la información de los registros es complementaria entre sí. Por último, se encuentran casos en los que existen discrepancias entre los mismos valores de diferentes registros, dando lugar a, por ejemplo, varias fechas de nacimiento para la misma persona o personas que en ocasionas registran solo su primer nombre y otras solo su segundo, o bien un mismo nombre pero con grafías distintas.

En un caso real, el comprador de la herramienta también puede indicar qué sistema de bases de datos tiene más fiabilidad, bien porque sea más moderno y/o los datos de ese sistema hayan sido actualizados recientemente o bien porque simplemente se guardaron de mejor manera en el momento de la carga. Esto permite crear una jerarquía de importancia en los sistemas cuando se efectúa el *match and merge*.

En este caso en particular, los tres sistemas de gestión de datos otorgados por el cliente, se encuentran en la máquina local del autor del trabajo, guardados como fichero CSV (*Comma-separated values*). Esto quiere decir que cada sistema tiene un directorio y en ese directorio se encuentran los ficheros de los datos. Un fichero equivale a una tabla, haciendo la carga de datos de forma más directa y eficaz.

La herramienta MDM ofrece la solución a todos estos problemas, gracias a al manejo de datos maestros.

4.3 Diseño del sistema

Cada sistema de gestión de datos puede tener una estructura distinta en cuanto a las relaciones que hay construidas entre las tablas y las entradas que se guardan en la base de datos. Por ejemplo, el sistema 1 puede tener en un mismo registro el tipo de vía y la dirección, mientras que en el sistema 2 únicamente está el tipo de vía y en el sistema 3 tipo de vía y dirección están separados en registros diferentes. Por eso, el trabajo del ingeniero es diseñar una estructura que pueda almacenar el modelo de datos, maximizando la calidad de los datos.

Normalmente, en un proyecto auténtico, el cliente expresa los atributos que desea guardar en el sistema *party model*, que es un tipo de esquema de entidades. La razón por la que la mayoría de proyectos MDM se definen con la estructura *party model* es porque, en vez de tener una tabla por entidad, agrupa las entidades similares bajo una misma tabla. El resultado es una base de datos extremadamente normalizada con muy pocas tablas, que se traduce en un acceso a los datos más sencillo y eficiente, donde las consultas (*queries*) son mucho más simples y con una sola llamada se puede obtener un gran volumen de datos. Esto ocurre gracias a que las consultas son más potentes, lo que agiliza el trabajo. Las actualizaciones de la base de datos utilizando el *party model* son mínimas o nulas, ya que el esquema nunca cambia. Esto hace que pueda experimentar un excelente incremento horizontal. Esto implica que una vez establecido el sistema, si en algún punto en el futuro se desea añadir un nuevo tipo de entidad, el modelo de datos absorbería esa entidad con la tabla que más compatibilidad tenga. Por ejemplo, si en el sistemas hay una entidad llamada «Persona» que guarda clientes particulares y en el futuro se quiere guardar información de comerciales, la entidad solamente necesitaría ajustes menores para poder adaptarse a este nuevo tipo de dato.

En cuanto a las desventajas, no es recomendable usar *joins* ya que al hacer la consulta se estarían recorriendo gran parte de los registros de la bases de datos y al rendimiento se vería afectado. No todo el mundo conoce este tipo de estructuras, lo que hace que al principio sea un desafío si no se han tratado previamente.

4.3.1. *Party Model*

El esquema del modelo de datos (4.1) se define con las siguientes entidades:

- **Coche:** Es el producto que venden los concesionarios, comprado por los clientes. Los atributos del coche son: La marca, el modelo, la matrícula, el color y el identificador con el que está asociado el coche al concesionario (es decir, la clave ajena).
- **Concesionario:** Es la entidad que contiene el nombre de los concesionarios y su propio identificador para que pueda ser referenciada. Se trata de una tabla padre, donde sus hijos son «Coche» y la tabla de relación entre «Concesionario» y «Persona».
- **Relación concesionario persona:** Esta tabla es la relación entre las personas que adquieren un vehículo y el concesionario que les vende el automóvil. Por eso, contiene dos claves ajenas, una apuntando hacia la entidad «Concesionario» y la otra apuntando hacia «Persona». Es un tipo de relación muchos a muchos, es decir, una persona puede haber comprado en varios concesionarios o en ninguno y un concesionario puede vender a muchos clientes o a ninguno.
- **Persona:** Es el eje central del modelo de datos. Esta entidad es donde se ejecutarán principalmente los procesos de emparejamiento y fusión de datos (*match and merge*). Incluye nombre, los dos apellidos, el número y letra del documento nacional de identidad, la fecha de nacimiento y la clave principal. Es la segunda y última tabla padre, referenciada por «Dirección», «Teléfono» e «Email».
- **Email:** Además de la dirección de *e-mail* (correo electrónico), en esta entidad se almacenan la clave ajena hacia «Persona» y un registro que se encarga de confirmar que la dirección de *e-mail* ha sido confirmada por el cliente y funciona de forma correcta. Si una dirección de correo electrónico ha sido confirmada por el cliente, esa dirección prevalecerá sobre las que no están verificadas durante el proceso de obtención de datos maestros provocado por la fase de fusión de datos. Cabe destacar la siguiente constricción: una persona solamente puede tener una única dirección de correo electrónico. Esta decisión de diseño será importante más adelante para escoger estrategias de *match and merge* (5.2.1).
- **Teléfono:** La entidad «Teléfono» es muy parecida a la de «Email». En este caso, almacena el atributo del teléfono y un valor que confirma si el teléfono ha sido verificado por el cliente, además de la clave ajena que apunta hacia «Persona». Al igual que *e-mail*, una persona solamente puede tener acceso a único teléfono.
- **Dirección:** Esta tabla contiene los atributos básicos para formar una dirección completa. El tipo de vía, la calle, numero de portal y puerta, provincia, código postal, ciudad y país. «Dirección» también es una entidad con la constricción de que una persona solamente puede tener una vivienda.

4.4 Diseño de los datos

En la siguiente lista se describe qué tipos de datos se han introducido al sistema y los errores que presentan. Estos conflictos son los que las herramientas Developer y MDM Hub tendrán que resolver para probar su rendimiento y fiabilidad. Las erratas a solventar son las siguientes:

- Registros repetidos tanto en un mismo sistema como entre distintos sistemas.

- Registros repetidos donde la información de las columnas está distribuida a lo largo de los sistemas. Por ejemplo, un mismo registro se repite en los sistemas 1 y 2 pero en el sistema uno falta el primer apellido, mientras que en el sistema 2 falta el segundo apellido. MDM Hub tendrá que fusionar ambos registros y no perder ninguno de los apellidos. Esto también se aplica si el registro está repetido en un mismo sistema.
- Eliminación de espacios en blanco sobrantes.
- Las entidades «Teléfono» e «Email» tienen el campo «Verificado», el cual es un *flag* que otorga más importancia cuando se encuentra activado. Se pretende que, cuando suceda la comparación entre dos números de teléfono o direcciones de *e-mail*, gane el registro que haya sido verificado previamente.
- Nombres de pila y nombres completos. Por ejemplo, MDM Hub tiene que comprobar que el nombre «Isa» e «Isabel» es el mismo y no separarlos en el proceso de *match*.
- En cuanto a los tipos de datos introducidos,

En las entidades «Persona», «Dirección», «Teléfono» e «Email» hay un total de ciento cuarenta y ocho registros (cuarenta y nueve procedentes de «SAP», cuarenta y tres de «Oracle» y cincuenta y seis de «Mongo»), de los cuales noventa registros son únicos. En las tablas «Concesionario» hay treinta registros en «SAP», treinta en «Oracle» y otros treinta en «Mongo», pero solo sesenta y tres de estos registros son únicos. En cuanto a la entidad «Coche», todos y cada uno de los automóviles son únicos, un total de seiscientos registros.

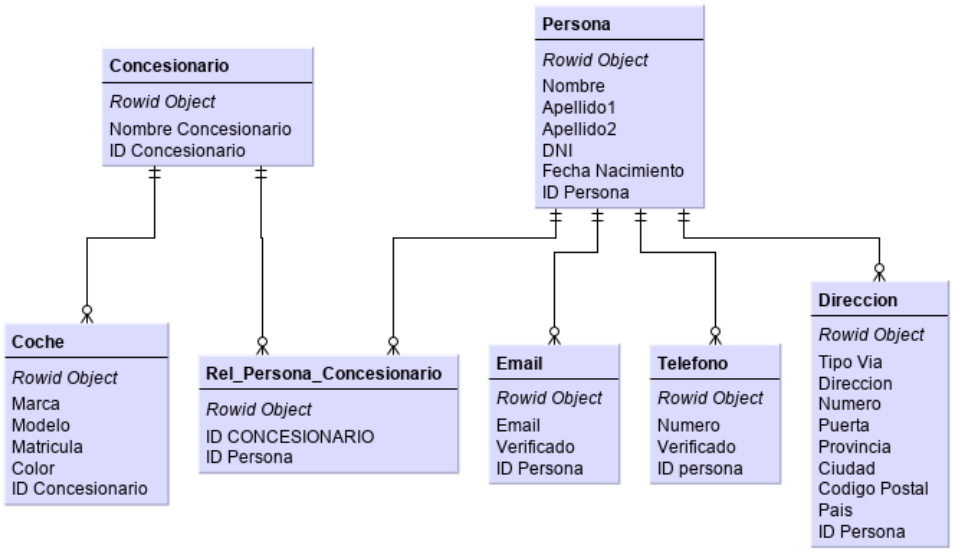

Figura 4.1: Esquema del modelo de datos (*party model*).

CAPÍTULO 5
Solución desarrollada

En capítulos previos se ha expuesto el contexto tecnológico en el que se ubica el proyecto y para qué son útiles las soluciones MDM, además del caso de estudio. A partir de este punto, se explica con detalle la arquitectura de la solución, así como qué aplicaciones se han utilizado, cómo y por qué se han utilizado, cuál es su objetivo y cómo interactúan entre ellas. En este capítulo se propone una solución MDM.

El capítulo está ordenado por herramientas para que el lector tenga claridad, tratando primero Developer y luego MDM. Sin embargo, ese no es el orden de uso para llegar a los datos maestros. Dicho orden es el siguiente.

- Una vez establecido con el cliente qué entidades van a transformar sus datos en datos maestros, se comienza con el diseño del esquema de modelo de datos (*party model*)
- Una vez diseñado, se crean las tablas de los objetos base, en MDM Hub, con la configuración de las columnas y las relaciones entre objetos para finalizar el *party model* (4.1).
- Establecido el esquema de modelo de datos y los objetos base, se crean las tablas Stage. Cada objeto base tendrá una o varias Stage dependiendo de la cantidad de sistemas de los que adquiera los datos.
- Pasando a Developer, se establecen las conexiones hacia los modelos de datos del cliente.
- Se extraen las tablas del cliente a Developer.
- Se crean y aplican las expresiones necesarias para la transformación de los datos sobre las tablas del cliente.
- Una vez estén limpios los datos, se cargan a las tablas Stage creadas en MDM Hub.
- En MDM Hub, los objetos base se cargan con toda la información almacenada en las Stage.
- Se aplica *match and merge* sobre los objetos base y la información se vuelve a almacenar sobre estos mismos.
- Los datos maestros están creados y se almacenan en los objetos base.

5.1 Informatica Developer

Informatica Developer es la herramienta ETL (*Extract Transform Load*) que se encarga de extraer los datos de las bases de datos del cliente (*Extract*), sin importar qué tipo de bases de datos sean o la forma en la que se almacenen. Una vez establecida la conexión con las bases de datos del cliente, puede empezar a aplicar transformaciones y expresiones (*Transform*) sobre los datos. Esto sirve como primera capa de limpieza de datos. Una vez homogeneizados los datos, son cargados (*Load*) en las tablas de Stage [5].

5.1.1. Extracción de los datos

El primer paso es acordar con el cliente la disposición de los datos, cómo y de dónde llega la información. Al ser un entorno de pruebas, los datos están almacenados de forma local. En un proyecto real este procedimiento implicaría conectarse a los sistemas del cliente mediante el *wizard* de conexión (5.1) definiendo los conectores como objetos de dato y ajustando los parámetros como tipo de datos, cabeceras o separadores entre otros. Una vez introducidos los datos de conexión se obtendrían las tablas con la información. En el caso de este proyecto la ubicación de los datos es el entorno local guardado, donde se disponen como planos de texto, con lo cual solo habría que especificar el directorio fuente.

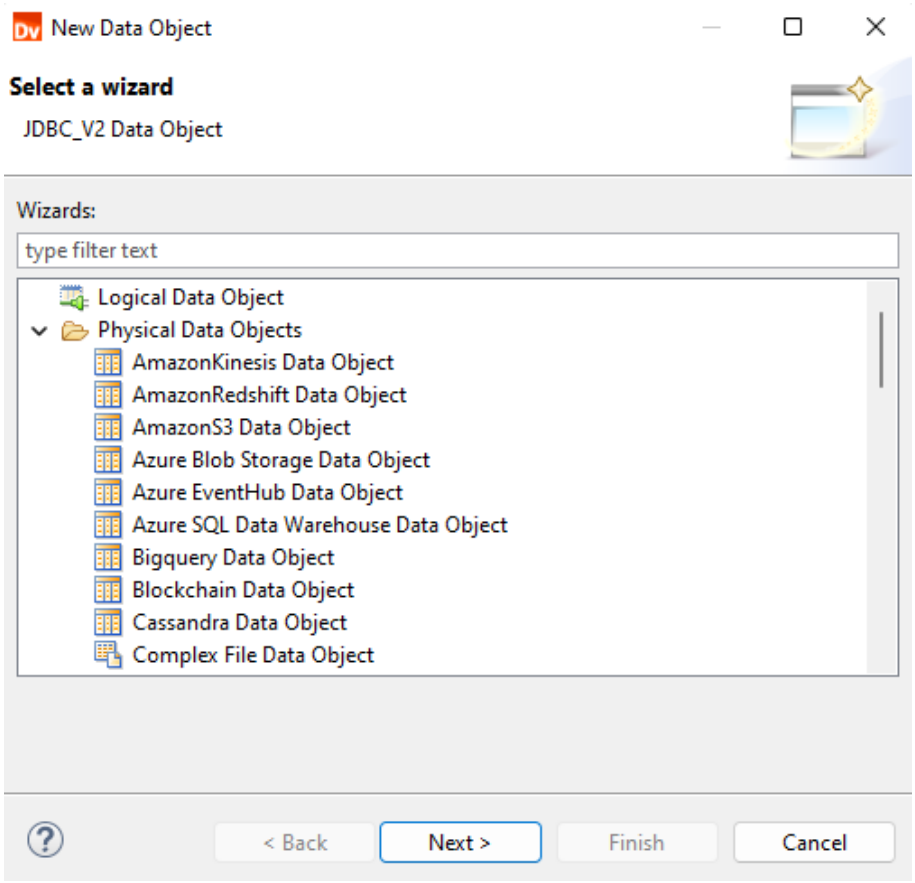

Figura 5.1: Interfaz de usuario de Developer para seleccionar una nueva conexión.

Al incorporar una fuente nueva al sistema, la capacidad de elegir la precisión de forma automática es limitada, por eso Developer pregunta al usuario de forma manual por la precisión de los registros, el tipo de dato para cada dato y en algunos, además, puede preguntar por una configuración más detallada. Por ejemplo, en los datos tipo fecha (*da-*

te) pregunta por el orden de los valores correspondientes a día, mes y año. También tiene la opción de importar la primera línea como nombre de la columna (5.2).

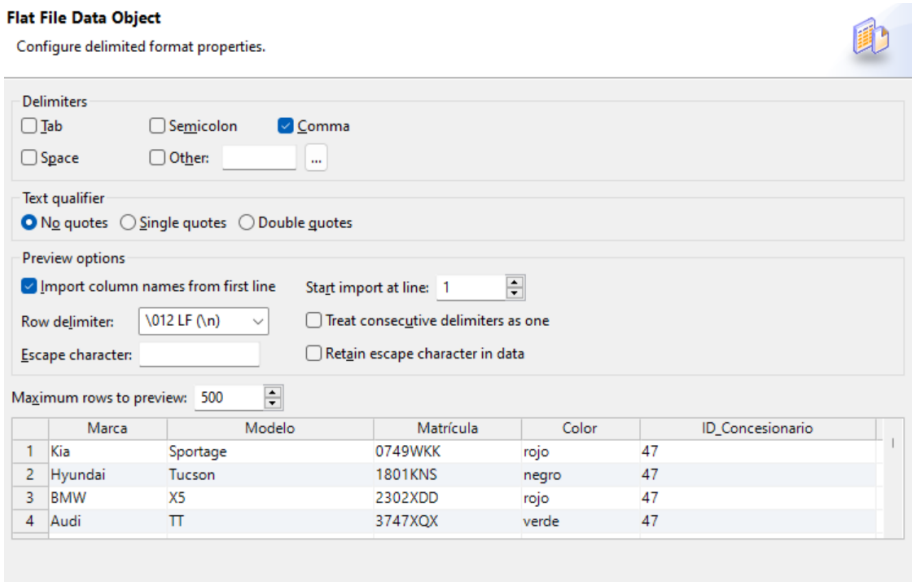

Figura 5.2: Configuración del importador de datos tipo CSV.

El término «precisión» se refiere a la cantidad de caracteres que puede tener un registro y depende en gran parte de los tipos de datos obtenidos del cliente. Por lo general suele suceder que los nombres, apellidos y direcciones tienen una precisión alta para evitar problemas de compatibilidad que pueda ocasionar que una persona cuente con un nombre largo o un apellido combinado, por ejemplo. También se tiene en cuenta de esta forma facilitar la inclusión de nombres extranjeros que pueden ocupar grandes cantidades de caracteres, cosa que no ocurre con los nombres locales habituales. Las fechas tienen una precisión mucho más ajustada porque siempre son de la misma longitud o la variación de longitud es mínima. Por estas diferencias hay que valorar cada campo y dejar un margen para poder evitar problemas en el futuro.

El proceso de extracción se repite por cada fichero. Son tres sistemas y cada sistema contiene siete entidades, en total suman veintiuna tablas (5.3) que hay que importar al sistema Informatica Developer.

5.1.2. Transformación de los datos

La segunda fase que implica un proceso ETL es el tratado de los datos. No se debe confundir esta etapa con el desarrollo de datos maestros. En la transformación de los datos se pretende hacer una limpieza en cuanto a la calidad del dato en sí, sin importar si está duplicado o si se va a descartar en el proceso MDM.

El proceso de transformación se enfoca principalmente en mapeo los datos. Esto consiste en señalar qué campos procedentes de las entidades son trasladados a las tablas Stage. Las tablas Stage toman su nombre del término inglés, traducido como «etapa», refiriéndose a que son parte de una fase intermedia en el proceso para obtener datos maestros. MDM usará los datos que provienen de las tablas Stage para la comprobación de coincidencia y la fusión de los datos (*match and merge*). Pueden existir una o más entidades Stage por cada objeto base modelado en MDM. Normalmente un sistema cliente equivale a un conjunto de tablas Stage. Con esto lo que sucede es que, por ejemplo, si en el sistema donde se importan las tablas existe la tabla «Coche» proveniente del sistema 1, entonces se creará una tabla Stage específica que almacenará los datos de la tabla «Co-

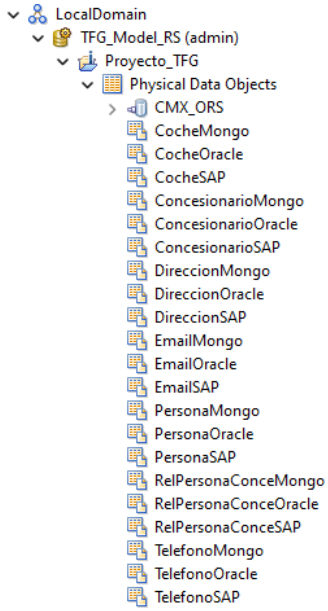

Figura 5.3: Todas las tablas guardadas en el sistema Informatica Developer.

che» del sistema 1. Las tablas Stage son creadas por la herramienta MDM después del modelo de datos. El proceso se explica en la siguiente sección (5.2).

Antes de construir los mapeos, es necesario definir algunas expresiones de limpieza de datos. Estas expresiones se llaman *mapplets* y funcionan como una función que se puede reutilizar en los *mappings*. La razón por la que es necesario añadir funciones de limpieza de datos es la siguiente: Si la fuente de datos es de baja calidad y contiene caracteres indeseables como espacios en blanco o tabulaciones, fallará el *match* porque identificará dos nombres que en la práctica son iguales como si fueran distintos, al tener uno un espacio en blanco y el otro no. Este error también aparece si hay personas con dos nombres y entre estos hay espacios innecesarios. La segunda razón por la que se necesitan *mapplets* es porque hay cuatro registros en las tablas Stage que necesitan ser poblados. Estos son **PKEY_SOURCE_OBJECT**, **SRC_ROWID**, **VERSION_SEQ** y **LAST_UPDATE_DATE**.

Los dos primeros son identificadores. La primera es la clave que MDM sobrescribirá como única más adelante y la segunda es la clave primaria de ese sistema de la información, usado para trazabilidad. El tercer atributo es la versión de la tabla, un campo técnico obligatorio cuyo fin es tener bajo control la diferentes versiones de datos de un mismo sistema. Por último, las tablas Stage necesitan la fecha en la que se ha ejecutado el *mapping*, un campo extremadamente importante para la resolución de conflictos en el *match and merge*. En este paso, cuando dos registros son posibles candidatos a ser dato maestro, MDM Hub escoge el más actualizado. En caso de que no exista ese campo, utiliza LAST_UPDATE_DATE. Como se menciona previamente, los *mapplets* son reutilizables, así que cada *mapping* contendrá estas dos funciones, que serán llamadas «Quitar_Espacios_En_Blanco» y «Version_Date». El primer *mapplet* presenta la siguiente expresión regular (*regex*).

```
RTRIM(LTRIM(REG_REPLACE(Input, '/s+', ' ')))
```

RTRIM y LTRIM se traducen en eliminar los espacios situados a derecha e izquierda respectivamente. REG_REPLACE es la expresión que sustituye todos los espacios en blanco (/s+) por un único espacio en blanco (' ') en la variable Input (entrada), tomando el Input como un registro de la tabla.

El segundo *mapplet* en comparación es más sencillo, siendo la versión (VERSION_SEQ) una constante que no será necesaria modificar en el futuro. La última fecha de actualización, que es una variable de sistema, se actualiza cada vez que se ejecuta el *mapping*.

VERSION_SEQ	'1'
LAST_UPDATE_DATE	SYSTIMESTAMP()

Una vez creados los *mapplets*, ya se puede proceder a traspasar información desde los objetos de datos del cliente a las tablas Stage de MDM.

La construcción de los mapeos en Informatica Developer ocurre de la siguiente forma:

- Comienza con la creación de una plantilla o lienzo en blanco de un *mapping*.
- En la plantilla en blanco se añade un objeto de datos. Este objeto de datos hace referencia a las fuentes que han sido importadas en el proceso de extracción (5.3). Para añadirlo simplemente hay que pinchar en el objeto y arrastrarlo al lienzo.
- Del mismo modo que se añade el objeto de datos, se añaden los *mapplets* al lienzo.
- En este punto es cuando se unen los puertos del objeto de datos con los *mapplets*. Se pretende que la información pase por la expresión regular encargada de eliminar los espacios en blanco sobrantes, así que cada puerto se une con un Input.
- El *mapplet* de versión y la última fecha actualizada utilizan variables de sistema y constantes que no tienen ninguna dependencia de los datos de entrada, pero hay que unir un puerto como entrada para que Developer entienda el flujo del *mapping*, aunque ese puerto no se use posteriormente. Por eso el puerto de entrada del *mapplet* Version_Date se llama *Dummy* (5.4), traduciéndose a falso o inútil.
- Los puertos de salida (*outputs*) se generan cuando los registros de los objetos de datos han sido asignados a los puertos de entradas de los *mapplets*. Los primeros se asignarán a la tabla de Stage.
- Para añadir una tabla Stage al lienzo hay que cliquear y arrastrar la tabla deseada del objeto de datos, y a continuación soltarla. Se debe comprobar que la tabla Stage añadida es la que está relacionada con el objeto de datos proveniente del cliente.
- Por último, se asignan los *outputs* de los *mapplets* a la entrada de las tablas Stage. Los registros VERSION_SEQ y LAST_UPDATE_DATE proceden del *mapplet* Version_Date, mientras que PKEY_SOURCE_OBJECT, SRC_ROWID, MARCA, MODELO, MATRICULA, COLOR, ID_CONCESIONARIO provienen del objeto de datos del cliente que ha pasado por la expresión que quita los espacios en blanco.

Así quedaría finalizado el mapeo de datos (5.4). Este proceso debería repetirse por cada tabla de cada sistema, necesitando un total de veintiuna repeticiones para conseguir la transformación total de los datos.

5.1.3. Carga de los datos

La exportación de los datos es la tercera y última parte de un proceso ETL. Consiste en cargar los datos que han pasado por los mapeos a las tablas Stage para que la información pase del cliente al dominio de MDM.

Para conseguir el envío de datos a las tablas Stage hay que ejecutar el mapeo, lo que se puede realizar una a una o bien creando un flujo de trabajo (*workflow*) que implemente todos los mapeos donde se ejecutará secuencialmente de forma automática. Este último es

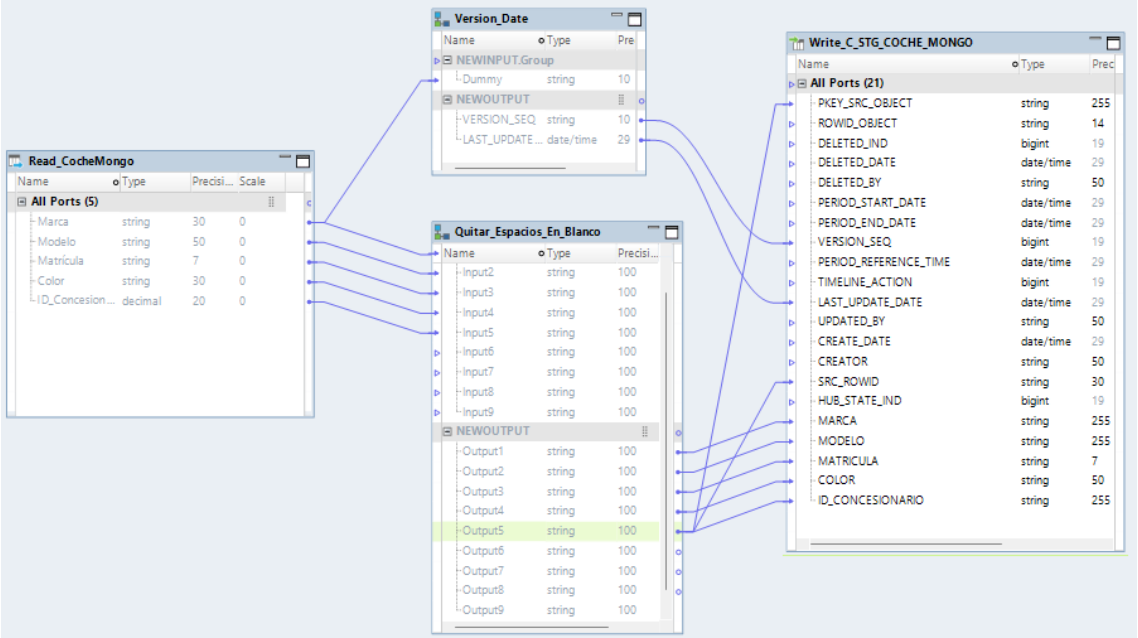

Figura 5.4: *Mapping* de la tabla «Coche» en Informatica Developer.

el procedimiento a seguir en un proyecto real. La ejecución de mapeos sin flujo de trabajo se utiliza para depurar y probar que el mapeo funciona de manera esperada. Una vez se ha confirmado que es correcto, se puede implementar el mapeo en el flujo de trabajo. Para ello, se debe añadir un *workflow* en el explorador de objetos una vez creado el flujo. Simplemente se arrastran los mapeos creados al lienzo y se unen en orden. La secuencia de ejecución es muy importante. Primero se incluyen los mapeos donde las tablas no tienen dependencia con ninguna otra entidad («Persona» y «concesionario»). El resto de tablas son dependientes de estas por las claves ajenas. Al no haber más dependencia de estilo padre e hijo, se pueden añadir los demás *mappings* al flujo de trabajo. «Dirección», «Rel-Pers-Conce», «Email», «Teléfono» y «Coche» (5.5).

Figura 5.5: Orden secuencial de los *mappings* en el *workflow*.

Una vez creado y configurado el flujo de trabajo, se procede a desplegarlo y ejecutarlo. El despliegue es sencillamente una propiedad a la que se accede cliqueando con el botón derecho y seleccionando *deploy*, creando así una aplicación que contiene el flujo de trabajo. La ejecución se ha de realizar en el Administrator de Informatica.

Mediante una conexión HTTPS, el siguiente paso es abrir el Administrator de Informatica.

`https://localhost:8443/administrator`

Administator es el centro de control de la herramienta Developer. En él se encuentran todos los servicios disponibles, como el repositorio de datos, el monitor de repositorio de modelos, la herramienta de análisis o el servicio de integración de datos. Este último es el

servicio que nos permite ejecutar aplicaciones que han sido desplegadas, como por ejemplo sería el *workflow* en este caso. Solo hay que dirigirse a las aplicaciones desplegadas e iniciar el flujo de trabajo como se indica en la figura 5.6.

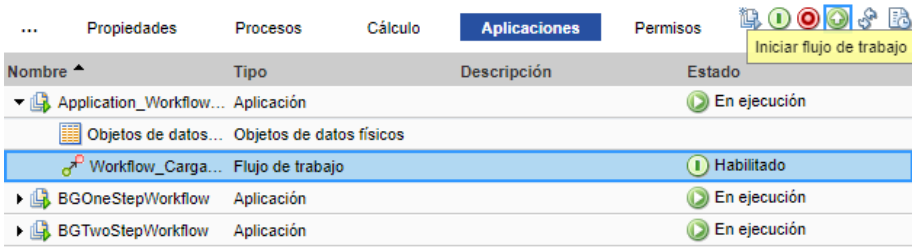

Figura 5.6: Flujo de trabajo habilitado y listo para ser ejecutado.

De este modo quedaría concluida la ETL. Todos los datos procedentes del cliente se han limpiado y guardado en las Stage y a partir de este momento el proceso de creación de datos maestros con MDM puede dar comienzo.

5.2 MDM Hub

La herramienta de *Master Data Management* que se va a utilizar es MDM Hub, diseñado también por la empresa Informatica, instalado sobre un servidor JBoss de Red Hat. Es la esencia de este trabajo de fin de grado y la encargada de hacer los registros maestros.

A continuación se detalla el proceso de configuración de la herramienta. El primer paso es conectar MDM Hub a la base de datos creada específicamente para ello, que estará almacenada localmente y será el producto final del proyecto. Una vez establecida la conexión a la base de datos es cuando se comienza a modelar la solución *party model* (4.1), creando los objetos base (*Base Objects*), que son: «Persona», «Concesionario», «Dirección», «Rel-Pers-Conce», «Email», «Teléfono» y «Coche». A estos objetos base se añaden las columnas y se comprueba que la precisión de los datos es la adecuada (5.7).

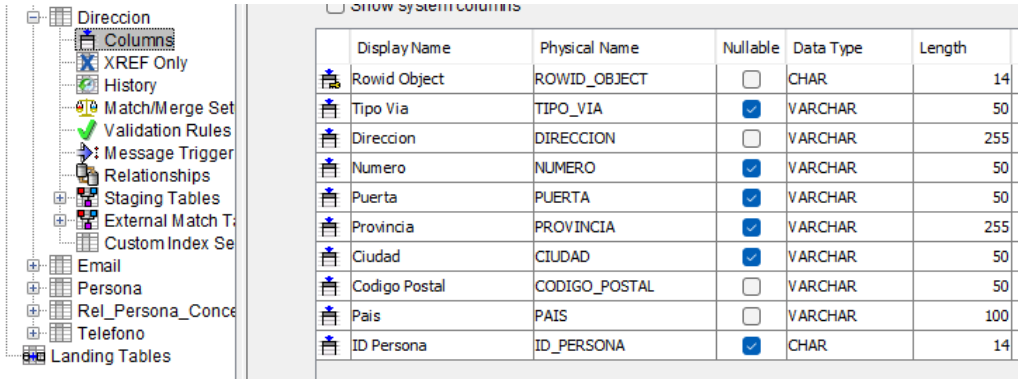

Figura 5.7: Interfaz MDM Hub, columnas de las entidad «Dirección».

Una vez establecidas las columnas del objeto base hay que crear las fuentes ERPs, que son los sistemas de donde va a proceder la información del cliente. En este caso son tres ERPs: «Oracle DB», «Mongo DB» y «SAP». Esto permite la creación de las tablas Stage. Cada objeto base se conforma a través de la fusión de los datos de tres objetos Stage, uno por cada sistema, con la misma cantidad de columnas que el objeto base. Las tablas Stage actúan como intermediario, recibiendo datos procedentes de las bases de datos del cliente. MDM Hub entonces ejecutará grupos de carga (*batch groups*) para poblar los objetos base.

Cuando se ejecutaron los lotes de carga, aparecieron varios problemas. El primero fue que las claves ajenas de los objetos hijo estaban mal configuradas y no se relacionaban, dando lugar a que los datos nunca llegaran a poblarse porque no encontraban la relación con el objeto padre. La solución para este problema fue realizar una limpieza total de las tablas Stage y comprobar, en el caso de las columnas del objeto base (5.7), que la clave escogida efectivamente era única y no era posible que estuviese vacía. Este proceso necesitó varias iteraciones para poder configurar un modelo perfecto. Después de arreglar el primer problema, apareció un segundo fallo consistente en que el atributo de la entidad «Persona» que contiene las fechas de nacimiento no se poblaba. Investigando este error se descubrió que Informatica Developer leía las fechas de la siguiente manera:

```
YYYY-MM-DD hh:mm:ss.ms
```

Mientras que las fechas de nacimiento dadas por el cliente seguían este otro patrón:

```
DD/MM/YYYY
```

Esta diferencia ocasionaba fallos de interpretación. La solución para este problema fue cambiar la configuración del intérprete de fechas de Informatica Developer. Tras esta modificación, aunque la información proporcionada por el cliente, por ejemplo, contenga el carácter «/» en vez de «-» y carezca de la hora de nacimiento, el intérprete cambia los caracteres e interpreta como cero (00:00:00) la hora, dando como resultado un valor tal que el mostrado en el siguiente ejemplo.

```
1965-02-14 00:00:00
```

Una vez resueltos los fallos de interpretación y relación de claves ajenas, se pueden ejecutar los lotes de carga para sobrescribir los fallos previos. Los lotes tienen que mantener un orden, parecido al del *workflow* de Developer. Primero se cargan las entidades padres que no tienen dependencia y después las entidades hijos, que están relacionadas hacia las padres.

Finalizada la ejecución de los grupos de carga, el resultado es que los objetos base están llenos con toda la información de los clientes, es decir, toda la información ha pasado de almacenarse en tres sistemas distintos a un único sitio: los objetos base. Este proceso puede hacer surgir la siguiente pregunta. ¿Qué pasa si se da el poco probable caso de que dos sistemas diferentes usan la misma clave primaria en las entidades clientes? La respuesta viene dada por el juego de las siguientes columnas:

- **PKEY_SOURCE_OBJECT:** Es la nueva clave primaria de los registros. MDM Hub se asegura de que esta sea única y no se repita. Aunque se haya poblado en el proceso de ETL, MDM Hub la sobrescribe.
- **SRC_ROWID:** Es la clave primaria que procede de la base de datos del clientes. Es la clave primaria original que, aunque deja de ser el quid, se mantiene en este nuevo registro.

5.2.1. *Match and merge*

Buscar coincidencias en los registros y fusionarlos es la función principal de MDM Hub. Como se viene comentando, esta herramienta es muy potente y personalizable, permitiendo pulir (*fine tunning*) hasta el más mínimo detalle la configuración para que el

proceso de *match and merge* sea perfecto, lo que resulta en una base de datos limpia y lista para ser explotada por la organización.

Cabe recordar que el proceso que se presenta es una recreación simplificada de un modelo real y por ello no se exhibirán todas las capas de complejidad y configuración que muestra la herramienta. Aun así, esto se discute en el capítulo de futuras mejoras (7).

Configuración

La variedad de opciones que presenta MDM Hub para llevar a cabo la fusión de registros es muy amplia. Por ejemplo, el tipo de estrategia puede ser exacta (*exact*) o difusa (*fuzzy*). La estrategia exacta se refiere a que los registros que se comparan tienen que ser idénticos, con sensibilidad incluso para diferenciar mayúsculas de minúsculas. Si dos registros coinciden significan que son duplicados.

La estrategia difusa es un tipo de regla más laxa, por lo que permite agrupar registros que se parecen. Por ejemplo, si se dan dos registros y uno contiene «IBM» y el otro «International Business Machines», MDM Hub configurado con estrategia difusa es capaz de darse cuenta de que se trata de la misma empresa, haciendo que estos registros se fusionen en uno.

Expuestas las dos estrategias, cada objeto base tendrá una diferente, por lo que hay que seleccionar la más adecuada para cada uno.

- **Persona:** Esta entidad tendrá una mezcla de estrategias. Cuando se comparan a dos personas, si coincide el DNI de ambas significa que es la misma persona, por lo que se aplica una estrategia exacta para ese campo. En el caso de no tener ningún valor en el campo del DNI, se puede comparar mediante el nombre y los dos apellidos, pero hay veces que las personas usan apodos o acortan su nombre, lo que daría problemas con una estrategia exacta. Por ejemplo, Paco y Francisco son el mismo nombre, así que en este caso usaríamos una estrategia de tipo difusa para poder agruparlo.

- **Dirección:** En este objeto base se ha dado la constricción a la entidad de que una persona solamente puede tener una única vivienda. En vez de aplicar la estrategia sobre la propia dirección en sí (si no coinciden las direcciones significaría que son dos personas distintas en vez de que pueda tener dos o más viviendas), se va a aplicar sobre la clave primaria del padre, es decir, el identificador de la persona. Al aplicarse sobre el identificador se interpreta que si son iguales, son la misma persona. Por esta razón se utiliza la estrategia exacta.

- **Concesionario:** Para comprobar que un concesionario es igual o diferente a otro simplemente habrá que comprobar si se llaman de la misma forma. Sin embargo, puede darse el caso de que aparezca «Autoventa» y «Autoventa S.L.». Aquí, para que el nombre del concesionario haga *match* y se puedan fusionar más adelante, habría que utilizar una estrategia difusa.

- **Coche:** Este objeto base usará una regla exacta muy simple: como los vehículos están dotados de matrículas, si dos coches tienen la misma matrícula, se trata del mismo automóvil.

- **Email:** Esta entidad sigue el mismo modelo de estrategia que «Dirección»: una persona solamente puede tener un único correo electrónico, y como se usa la clave primaria del objeto base «Persona», la mejor estrategia de *match* es exacta.

- **Teléfono:** Siguiendo con la constricción de que una persona solamente puede tener un único teléfono, lo más sencillo para conseguir *matches* sería seguir la misma estrategia que en el caso de «Email», es decir, una estrategia exacta sobre la clave ajena que apunta hacia la entidad padre «Persona».
- **Relación persona concesionario:** Por último, para la tabla de relación se aplicaría la estrategia exacta ya que está entrelazada con claves ajenas que son identificadores únicos.

Una vez elegidas las reglas de cada objeto base, se seleccionan las columnas por las que se irán comparando los registros.

- **Persona:** La regla de este objeto base es la más complicada puesto que combina dos estrategias y cada estrategia usa un conjunto de reglas diferentes. El primer conjunto es el de la estrategia difusa, que usará las columnas «Nombre», «Apellido1» y «Apellido2» combinado con la dirección «Tipo vía», «Dirección», «Número» y «Puerta», resultando en que si una persona se llama igual que otra y además tienen la misma dirección, se interpreta que se trata de una única persona.

  El segundo conjunto de reglas sigue la estrategia exacta, para ello requiere la columna «DNI». Si varias personas tienen el mismo documento de identidad, son la misma persona.
- **Dirección:** Esta entidad es la que más columnas presenta, nueve. Únicamente una de ellas es necesaria, que es «ID Persona», gracias a la constricción de que una persona solo puede tener una vivienda. Con encontrar dos personas con el mismo valor en esta columna, se interpreta que su dirección es la misma.
- **Concesionario:** La columna elegida para esta entidad es el «Nombre Concesionario» porque sigue una estrategia difusa y solamente necesitará un único conjunto de reglas.
- **Coche:** En esta tabla solo es necesaria una columna, «Matrícula», que funciona como identificador único de los automóviles.
- **Email:** En esta tabla se usará la clave ajena que apunta a persona, «ID Persona».
- **Teléfono:** Al igual que «Email», el objeto base de «Teléfono» se une con el registro que apunta a la entidad padre, que es «ID Persona».
- **Relación persona concesionario:** Esta entidad solamente tiene dos columnas, ambas estando relacionadas con los objetos base padre. Las dos columnas «ID Persona» y «ID Concesionario» serán necesarias para la comparación de los registros.

Este tipo de configuración hace que se planteen ciertas preguntas, como: ¿Qué pasa si una persona tiene dos viviendas distintas?

Aunque exista la constricción de que una persona solo tiene una vivienda, eso no significa que la persona no haya introducido direcciones diferentes en distintos sistemas. Por ejemplo, cuando un persona compró un coche hace años y otro ahora, es responsabilidad de MDM Hub escoger una sola dirección, o bien la que se guardó hace años o bien la más reciente. El programa está configurado para escoger el registro más actualizado. Gracias a esto, en el caso de conflicto consistente en que una persona aparentemente tenga dos o más registros de dirección distintos, MDM Hub guardará el más nuevo. Escoger el registro más reciente funciona en todas las entidades que tienen la constricción de tener un único registro, es decir, en las entidades «Dirección», «Email» y «Teléfono».

El siguiente nivel de configuración para el *match and merge* son las reglas de validación. Estas reglas utilizan las propias columnas de los objetos base para discernir una solución.

Este proyecto se ha diseñado para que dos de los objetos base («Teléfono» e «Email») tengan reglas de validación gracias a la columna que ambos tienen llamada «Verificación». Dicha columna almacena el valor 1 ó 0, siendo 1 que sí está verificado y 0 que no ha sido verificado. Funciona de la siguiente manera: Si una persona tiene dos teléfonos distintos, uno en cada sistema, y uno de ellos tiene el *flag* de verificado, este será el registro victorioso cuando se lleve a cabo el *merge*. En cuanto al funcionamiento técnico, lo que se aplica es que si el *flag* de verificado es 0, el registro de ese número pierde credibilidad (5.8). La credibilidad es un porcentaje que hace que a la hora de fusionar registros siempre gane el que tenga más credibilidad.

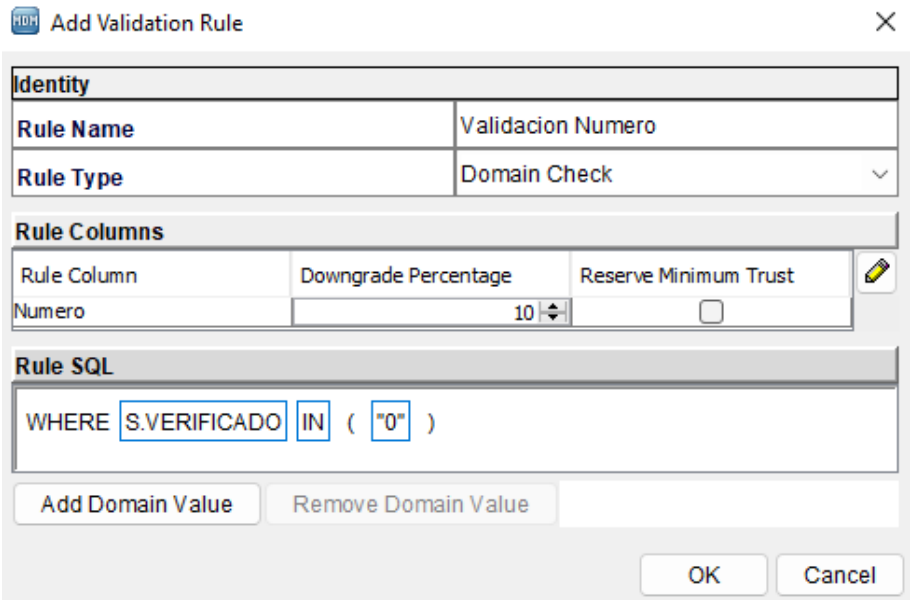

Figura 5.8: Interfaz MDM Hub, ajustando la regla de validación de número.

El uso de esta reglas de validación es muy útil, pero introduce dos nuevas flaquezas. En primer lugar, ¿qué pasaría si un registro de «Teléfono»/«Email» que fuese nulo tuviese, sin embargo, el *flag* de verificación activo?

En dicho caso, entre un registro nulo que está verificado y un registro con un número funcional que no ha sido verificado, ganaría el nulo. Para prevenir este tipo de situaciones la solución sería crear una regla de validación extra. En el caso de encontrar un teléfono nulo, este registro perdería porcentaje de credibilidad (5.9). Esta regla también se aplica a «Email».

¿Y qué pasaría si un *flag* de verificado fuese nulo pero el otro *flag* de verificación estuviese desactivado?

Como en la solución previa, en este caso se resolvería la situación creando una nueva regla de validación. En este caso, la regla creada haría bajar el porcentaje de credibilidad del registro que es nulo.

Ejecución

Una vez establecidos todos los parámetros, es el momento de lanzar a ejecución el *match and merge*. El orden de ejecución es de suma importancia. Primero hay que ejecutar las entidades padre «Persona» y «Concesionario», que no tienen ninguna dependencia

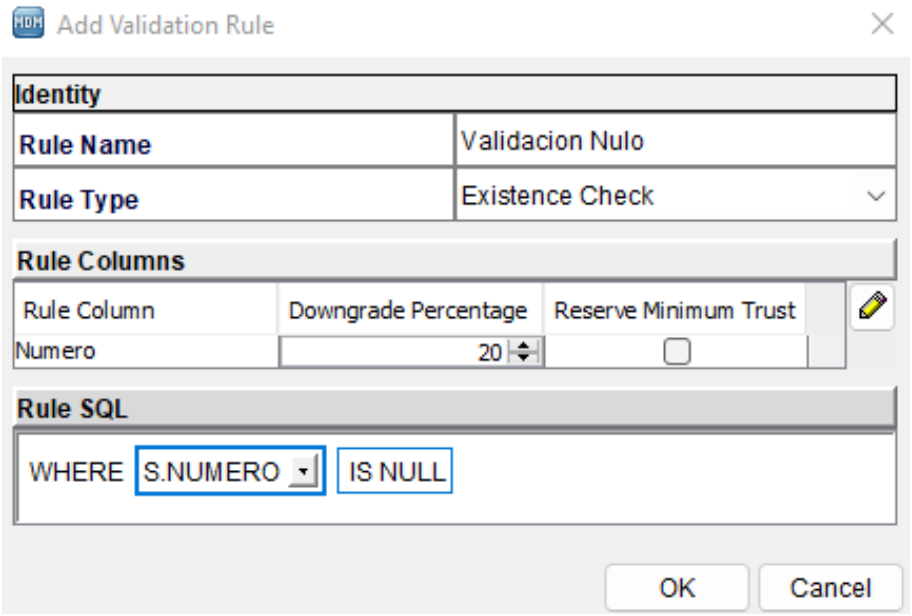

Figura 5.9: Interfaz MDM Hub, ajustando la regla de validación.

hacia los hijos, y después se ejecutan las tablas que tienen una clave ajena apuntando hacia las entidades padre. Para ejecutarlas hay que crear un grupo de carga, similar a los lotes creados para cargar de las tablas Stage a los objetos base, pero esta vez se ejecutará la configuración que se ha establecido para el *match and merge*.

Lanzada la ejecución, el uso de la herramienta MDM quedaría finalizado. El resultado se guarda en los objetos base y solo quedaría comprobar que las bases de datos han sido pobladas de manera correcta.

Al concluir este proceso, MDM Hub finalmente ha creado los datos maestros.

CAPÍTULO 6
Resultados experimentales y discusión

Tras haber expuesto todo el proceso de trabajo llevado a cabo para la gestión de los datos de la hipotética empresa de concesionarios, en esta sección se procede a discutir los resultado obtenidos y comentar la fiabilidad y el éxito de las herramientas empleadas.

La mayoría de los cambios introducidos han surtido efecto y funcionan como se esperaba de acuerdo a los objetivos propuestos. En el capítulo 7 se profundiza sobre el grado de superación de estos. En este capítulo se detallan en concreto los resultados de los conflictos de casos de uso expuestos en el capítulo 4. Por la extensión de la información simulada introducida para observar el funcionamiento de la metodología desarrollada, se exponen solo algunos de los resultados obtenidos con el fin de ejemplificar todas las situaciones de conflicto con esa misma resolución tras la ejecución del trabajo.

Analizando entidad por entidad, todos los sistemas en las entidades «Persona», «Dirección», «Teléfono» y «Email» suman un total de ciento cuarenta y ocho registros. Cuarenta y nueve registros procedentes de «SAP», cuarenta y tres de «Oracle» y cincuenta y seis de «Mongo». Cada sistema tiene treinta registros únicos, es decir, en total hay noventa registros únicos, y cincuenta y ocho de los registros están repetidos.

El objeto base tiene un total de noventa registros únicos, lo que significa que la herramienta MDM Hub ha eliminado todos y cada uno de los duplicados. En cuanto a la calidad de datos almacenados, todas las columnas de «Persona» están pobladas, a excepción de cinco registros de la columna «Fecha de nacimiento» y dos registros ausentes en la columna de «Apellido2». Esta falta de datos no es error de MDM Hub, sino que nunca se llegó a poseer esa información, ni siquiera en el sistema del cliente. Con este resultado podemos comprobar que MDM Hub es una herramienta extraordinaria. A continuación se exponen algunos ejemplos de los resultados en los que se apoya esta afirmación.

En el primer ejemplo se puede observar que ninguno de los tres registros repetidos provenientes del cliente estaban completos. MDM Hub ha adquirido toda la información y la ha unificado de forma correcta.

Sistema	Nombre	Apellido1	Apellido2	DNI	Fecha nacimiento
SAP	″	″	'Armando'	'14947634A'	'14/03/1979'
Oracle	″	'Betania'	″	'14947634A'	'14/03/1979'
Mongo	'Nora'	″	″	'14947634A'	'14/03/1979'
MDM Hub	'Nora'	'Betania'	'Armando'	'14947634A'	'1979-03-14 00:00:00'

El siguiente ejemplo ilustra el uso de las reglas de validación, en concreto cómo, de entre dos registros, se escoge el número de teléfono que sí ha sido validado.

Sistema	ID Persona	Número	Verificado
SAP	'953627618'	'735065751'	'1'
Mongo	'230'	'111111111'	'0'
MDM Hub	'953627618'	'735065751'	'1'

El último ejemplo muestra la capacidad de MDM para completar registros solventando el uso de apócopes, a pesar de que este registro conlleve más información que el registro rival al incluir un segundo apellido que el rival no aporta.

Sistema	Nombre	Apellido1	Apellido2
SAP	'Eduardo'	'García'	''
Oracle	'Edu'	'García'	'Sastre'
MDM Hub	'Eduardo'	'García'	'Sastre'

Sin embargo, se observa que no todos los resultados son favorables. Por ejemplo, en el caso de una hipotética Alicia, se encuentran dos direcciones de correo diferentes. Las reglas de validación no han funcionado correctamente porque ha prevalecido la dirección de correo incorrecta, es decir, la dirección no verificada ha sido escogida en lugar de la que sí estaba verificada. Este error se repite en dos registros de «Email» y en ocho de «Teléfono».

Sistema	Email	Verificado
SAP	'alicia@google.com'	'1'
Mongo	'aaaaaaaaaa@google.com'	'0'
MDM Hub	'aaaaaaaaaa@google.com'	'0'

Hay varios motivos por los que puede suceder este fallo, todos de ellos relacionados con la configuración de MDM Hub. Comenzando por las propias reglas de validación como origen del problema, aumentar el porcentaje de degradación cuando se encuentra la columna Verificado con el *flag* a 0 ayudaría a la resolución del error. El segundo motivo podría ser una incorrecta configuración de las estrategias de *match and merge*. La tercera opción es que sea causa de un incorrecto uso de claves ajenas y/o relaciones entre tablas. Por último, es posible que lo que suceda sea que un tipo de sistema tenga más posibilidades de ser aceptado, aplicándose así a todos los registros que tiene ese sistema.

CAPÍTULO 7
Conclusiones y trabajo futuro

Partiendo de los resultado obtenidos y expuestos en los puntos anteriores, se extraen las conclusiones a continuación expuestas sobre el uso de herramientas MDM para proyectos de gestión de datos de naturaleza similar al ejemplo realizado.

7.1 Futuras mejoras del proyecto

MDM es una herramienta que admite una cantidad abrumadora de configuraciones que van desde el propio servidor de instalación al sistema operativo donde se ejecuta MDM, entre otros. Esta versatilidad permite su aplicación para todo tipo de proyectos, lo que la convierte en una herramienta muy flexible y útil. Si además se combina con una ETL, como se ha expuesto en este proyecto, se potencia su capacidad de customización y gestión eficaz de datos.

Se deberían aplicar algunos cambios para mejorar la calidad de este proyecto. Habría que buscar un modo de conseguir replicar de forma más fiel un proyecto real. Si se hubiese conseguido mas fidelidad, se hubiese observado en los resultados una mayor potencia de la herramienta Developer y MDM. Si esta metodología se hubiera llevado a cabo para el proyecto de un cliente más ambicioso, lo que conllevaría más peticiones específicas y mucha más personalización, habría que crear más reglas para cubrir estas necesidades planteadas, tanto de estrategias de *match and merge* como un uso más intensivo de la herramienta ETL.

7.1.1. Informatica Developer

En cuanto a mejoras para la herramienta Developer (ETL), hay que tener en cuenta que esta permite un total de cincuenta y tres tipos de conexiones diferentes. En lo que a transformaciones se refiere, contiene treinta y tres tipos diferentes de funciones, de las cuales cabe destacar comparaciones, filtros, uniones y usos de *scripting* tanto de Python como de Java. Un hipotético caso en el que realizando una mejora se apreciaría esta potencia comentada sería el siguiente: En una transformación se reciben dos objetos de datos del cliente que provienen de sistemas distintos. De estos objetos se requieren datos de ambos. Pasando por la función de unificación, se obtendría una única tabla, la cual podrá ser filtrada para que únicamente almacene datos entre dos fechas escogidas, finalmente exportando todo la información a la tabla Stage.

Otras mejoras que se pueden aplicar a la ETL Developer en la etapa de carga (*Load*), sería añadiendo «tareas» en el lienzo del *workflow*. Aunque hay varios tipos de tareas disponibles, sobresalen dos por su utilidad para este propósito concreto. La primera es la

tarea de notificación. Mediante una configuración de STMP, esta permite enviar un correo electrónico cuando el flujo de trabajo esté siendo ejecutado. Además, como el flujo de trabajo permite crear varias rutas en función de los resultados obtenidos en los *mappings*, esto significa que si un *mapping* da error, se puede notificar mediante la tarea de notificación. La segunda tarea que permitiría una mejora sustancial es la tarea de comando. Esta permite ejecutar un *script*. En este caso, al ser Windows, sería *Shellscript*, creado por el usuario. Una vez que se ha establecido toda la configuración de MDM hub y de Developer y ha sido demostrado que funciona, es posible crear esta tarea de comando que ejecuta todos los grupos de carga. La tarea de comando contendría los grupos de carga de Stage a objeto base y los grupos de carga de *match and merge*. En consecuencia, el flujo de trabajo sería la única aplicación que es necesaria ejecutar para que se efectúen todos los cambios en el sistema.

7.1.2. MDM Hub

En cuanto a MDM Hub, si se quisiera obtener mayor potencia de tratamiento de datos, se podría aprovechar que MDM permite ser usado en un clúster con múltiples nodos para un procesamiento más rápido. Esta opción es especialmente útil cuando el cliente dispone de millones de registros, que en casos reales es lo que sucede habitualmente y no una excepción.

La herramienta MDM Hub también dispone de ETL. Esta herramienta es más sintética que Developer. Funciona de la siguiente forma: MDM cuenta con una serie de tablas *Landings*, del término inglés para «aterrizaje». En estas tablas *Landings* se podría cargar los datos de los clientes sin usar Developer. Una vez cargada toda la información en las tablas *Landings*, habría que crear un *mapping* en MDM Hub que conectara todos los registros que se desea usar a las tablas Stage. A partir de aquí el proceso es igual que como se ha mostrado a lo largo del proyecto. Estos *mappings* también tienen, aunque más simples, funciones de transformación.

Otra cualidad a destacar de MDM Hub es el *trust*, o confianza en castellano. Es una función específica para reducir el porcentaje de confianza con el paso del tiempo. Normalmente usado cuando se quiere hacer una transición de sistema, el sistema que se desea desechar, perdería confianza a lo largo del tiempo, haciendo que los registros del nuevo sistema se mantengan.

ActiveVOS

Con la instalación de MDM Hub se añade el programa ActiveVOS. No es necesario instalarlo pero evita problemas de dependencia. Si en un futuro se quisiera ampliar la utilidad del conjunto de programas instalados, sería un instrumento de gran utilidad.

Esta herramienta es un sistema de orquestación de procesos [6], es decir, que permite automatizar procedimientos. Esto ocurre mediante la creación de un flujo de trabajo, pero en lugar de utilizar *mappings* como unidades de tarea (como en Developer), se emplearían ejecuciones de programas como unidades de tareas. Cabe destacar que ActiveVOS puede usar llamadas HTTPS.

Cleanse Server

Otro componente opcional en la instalación de MDM es el *Cleanse Server*. Esta herramienta adicional incluye funciones para la herramienta de *mapping* de MDM Hub, como pueden ser *Is Integer Null*, *Format Date* o *Max Float*, entre muchas otras expresiones.

Otra función extremadamente importante de *Cleanse Server* va destinada a la configuración del motor de búsqueda inteligente, o *smart search* [7] en inglés. La principal característica de este tipo de búsqueda es la utilización de índices distribuidos. También permite hacer búsquedas a través de llamadas HTTPS.

Resource Kit

Por último, el equipo de recursos. Como su propio nombre indica, este equipo consiste en ejemplos de esquemas, muestras de datos, utilidades de servicios y configuraciones básicas entre otras. La intención detrás de la utilización del *resource kit* es que proporcione una integración del MDM Hub más rápida, para así poder comprobar la configuración con los ejemplos dados por el equipo de recursos.

7.2 Conclusión

Los resultados obtenidos en el capítulo previo (6) permiten llegar a la conclusión de que MDM Hub, en combinación con Informatica Developer, proporciona una herramienta extremadamente potente. La clave para conseguir unos resultados favorables reside en la correcta configuración de la herramienta, ya que el proceso de ejecución es semiautomático. Este tipo de producto es muy llamativo para las empresas por el hecho de que, una vez instalado, su uso es simple y directo, proporcionando unos resultados excepcionales en materia de datos maestros.

7.3 Opinión personal

Los objetivos de este trabajo se resumían en mostrar el potencial que tiene la herramienta MDM Hub y cómo funciona. Los resultados obtenidos sí que muestran la capacidad de MDM Hub de obtener datos maestros en combinación con Developer. A pesar de ello, me hubiera gustado obtener un resultado más refinado en el que no hubiese errores en lo que a las reglas de validación de datos se refiere, ya que esto afectó a la calidad de los datos en las entidades «Email» y «Teléfono». Si hubiera alcanzado tal grado de eficacia, habría superado mis expectativas, demostrando de forma impecable que MDM Hub es una herramienta indispensable y óptima en lo que a gobernanza y limpieza de datos se refiere.

Un cambio que añadiría si siguiese refinando este sistema sería utilizar unas bases de datos de cliente que no fueran tan parecidas al sistema de objeto de base, es decir, añadir más entidades en la bases de datos clientes para que la variación existente entre sistemas de clientes se incrementase. Esto hubiera complicado el proceso de fusión de datos, con lo que hubiera sido posible exponer más capas de capacidad que tiene MDM Hub y cómo hubiese podido resolver eficazmente otros conflictos más avanzados.

Dicho esto, estoy contento con los resultado obtenidos. Creo que se trata de un proyecto diferente con respecto a lo que se ve habitualmente en la Escuela Técnica Superior de Ingeniería Informática y considero que he aprendido muchas disciplinas distintas a través de él, como el uso e importancia de las ETL y todo el proceso de obtención de datos maestros, desde la instalación de la herramienta, pasando por su configuración y uso, hasta la solución de datos maestros del cliente que finaliza dicho proceso.

CAPÍTULO 8
Anexo: Objetivos de desarrollo sostenible

OBJETIVOS DE DESARROLLO SOSTENIBLE

Grado de relación del trabajo con los Objetivos de Desarrollo Sostenible (ODS).

Objetivos de Desarrollo Sostenible	**Alto**	**Medio**	**Bajo**	**No procede**
ODS 1. **Fin de la pobreza.**				X
ODS 2. **Hambre cero.**				X
ODS 3. **Salud y bienestar.**				X
ODS 4. **Educación de calidad.**				X
ODS 5. **Igualdad de género.**				X
ODS 6. **Agua limpia y saneamiento.**				X
ODS 7. **Energía asequible y no contaminante.**				X
ODS 8. **Trabajo decente y crecimiento económico.**		X		
ODS 9. **Industria, innovación e infraestructuras.**		X		
ODS 10. **Reducción de las desigualdades.**				X
ODS 11. **Ciudades y comunidades sostenibles.**				X
ODS 12. **Producción y consumo responsables.**				X
ODS 13. **Acción por el clima.**				X
ODS 14. **Vida submarina.**				X
ODS 15. **Vida de ecosistemas terrestres.**				X
ODS 16. **Paz, justicia e instituciones sólidas.**				X
ODS 17. **Alianzas para lograr objetivos.**				X

La naturaleza de este proyecto no contempla inicialmente la sostenibilidad como un punto a perseguir entre sus valores, puesto que se trata de un proyecto exclusivamente dedicado a ayudar a empresas y organizaciones en materia de creación de datos maestros. También se debe a que este proyecto se inicia con anterioridad a la inclusión de estos objetivos como parte esencial de todos los Trabajos de Fin de Grado presentados en la Universidad Politécnica de Valencia.

Dicho esto, el trabajo, al centrarse en mejorar la gestión virtual de datos, se dedica de forma indirecta a mejorar la infraestructura tecnológica del sector empresarial (aunque aplicable a cualquier tipo de organización que cuente con registros de clientes o bases de datos) mediante la innovación informática. A fin de cuentas, que el principal propósito sea desarrollar una solución MDM es posible por la creciente demanda de mejoras tecnológicas en todo tipo de sectores. Precisamente esto se comenta en el resumen inicial del trabajo. Sin la labor de los ingenieros informáticos y la creación de este tipo de soluciones, sería mucho más difícil hacer frente a los retos logísticos de las empresas.

Suplir esta demanda técnica es de vital importancia, puesto que con ello se evitan muchos problemas derivados que no es necesario que aparezcan. Aligerar y optimizar los trámites virtuales ahorra tiempo, recursos humanos y operaciones innecesarias que se desperdician y se traducen en pérdida de recursos energéticos por ejemplo. Toda mejora responsable es bienvenida y suma en la carrera hacia un desarrollo sostenible.

Además, al trabajar para aumentar la capacidad de acción de empresas y organizaciones, que el proyecto esté sujeto a los objetivos de desarrollo sostenible queda determinado, parcialmente aunque en gran medida, por los propios objetivos de dichas empresas y organizaciones. El potencial sostenible inicial de la herramienta desarrollada por el trabajo puede acabar traduciéndose de múltiples maneras. Si la empresa que hace uso de la herramienta es una empresa que se ciñe a valores respetuosos con el medioambiente y el futuro de la sociedad, la capacidad de optimización que le aporta la herramienta le permitirá llegar más lejos con estos objetivos y ser más sostenible aún. Sin embargo, también existe la situación opuesta en la que la empresa no tiene en absoluto en cuenta este tipo de valores. En este caso, será difícil que la herramienta ayude a que se agilice la toma de decisiones respetuosas, ya que no hay una intención por parte del usuario de utilizar la herramienta para facilitar el bien común.

Esta variabilidad de resultados dependiente de la moralidad con la que se aplica la herramienta es imprevisible y se encuentra fuera del control del creador de la solución. Sin embargo, la herramienta es neutra y no se promueve que se utilice con fines explotativos ni irresponsables. En este caso, el potencial para su buen uso por todas las facilidades que otorga la herramienta es más importante que tratar de remediar las políticas empresariales del cliente. Eso sería labor de otro tipo de profesionales y sectores a los que sin duda se anima a que busquen soluciones para evitar este tipo de comportamientos abusivos que tristemente se ven a diario en el mundo empresarial. Desde el proyecto se espera que su implementación pueda formar parte de un conjunto de cambios hacia el desarrollo sostenible.

Bibliografía

[1] B. Otto and M. Ofner, "Strategic business requirements for master data management systems," vol. 2, 08 2011.

[2] S. Sadiq, X. Zhou, and M. Orlowska, "Data quality - the key success factor for data driven engineering," in *2007 IFIP International Conference on Network and Parallel Computing Workshops (NPC 2007)*, 2007, pp. 48–56.

[3] "Gdpr," https://gdpr-info.eu/, Última vez accedido: Junio 19, 2022.

[4] Informatica, "Installation guide for microsoft sql server with red hat jboss 10.4," https://docs.informatica.com/master-data-management/multidomain-mdm/10-4-hotfix-2/installation-guide-for--microsoft-sql-server-with-red-hat-jboss/preface.html, Última actualización: Febrero 01, 2022.

[5] "Staging data," https://en.wikipedia.org/wiki/Staging_(data), Última vez accedido: Junio 20, 2022.

[6] Informatica, "Activevos," https://docs.informatica.com/es_es/master-data-management/multidomain-mdm/10-4/guia-de-seguridad/seguridad-a-nivel-de-aplicacion/activevos.html, Última vez accedido: Junio 27, 2022.

[7] Shay Banon, "Elasticsearch," https://github.com/elastic/elasticsearch, Última vez actualizado: Junio 14, 2022.

[8] Informatica, "Installation and configuration guide, data integration hub 10.2," https://docs.informatica.com/data-integration/data-integration-hub/10-2/installation-and-configuration-guide/preface.html, Última actualización: Agosto 14, 2020.

[9] Microsoft, "Sql server downloads," https://www.microsoft.com/es-es/sql-server/sql-server-downloads, Última vez accedido: Junio 20, 2022.

[10] Informatica, "Create an operational reference store," https://docs.informatica.com/master-data-management/multidomain-mdm/10-4-hotfix-2/installation-guide-for--microsoft-sql-server-with-red-hat-jboss/hub-store-installation/create-an-operational-reference-store.html, Última vez accedido: Junio 20, 2022.

[11] ——, "Import the metadata into the operational reference store," https://docs.informatica.com/master-data-management/multidomain-mdm/10-4-hotfix-2/installation-guide-for--microsoft-sql-server-with-red-hat-jboss/hub-store-installation/import-the-metadata-into-the-operational-reference-store.html, Última vez accedido: Junio 20, 2022.

APÉNDICE A
Manual de instalación

Este capítulo presenta el procedimiento necesario de instalación de las herramientas que se han usado para llevar a cabo este trabajo de fin de grado.

Para implementar una solución MDM es necesario extraer los datos del cliente, aplicar un filtro o expresión que ayude a depurar y mejorar la calidad de datos inicial, ejecutar el *match and merge* de los datos y por último guardar esos datos.

Las herramientas necesarias para llevar este proceso a cabo son bases de datos y cliente de bases de datos, para guardar los datos del cliente durante y al finalizar el proceso. La herramienta ETL sirve para la carga inicial de datos, tanto del cliente como del servidor. Cliente MDM resolverá los casos de datos duplicados y realizará la homogeneización de la totalidad de los datos, además del servidor donde se almacena la aplicación MDM.

La guía de instalación de Informatica [4] muestra en profundidad cómo instalar el software de diferentes formas y con diferentes configuraciones según el fin que se desee. A continuación se presentan los pasos detallados para la instalación del entorno utilizado en este trabajo de fin de grado.

A.1 Instalación de Informatica Developer

A.1.1. Preinstalación

La instalación se ha llevado de forma local en un ordenador proporcionado por Infoverity. Los requerimientos mínimos de Informatica para la instalación y correcto funcionamiento de MDM son 4.9 GB de espacio de disco y 8 GB de RAM [4].

A estos requerimientos hay que añadir la herramienta de integración de datos (IDQ), que requiere 8GB de RAM, 4 núcleos de CPU y 8 GB de disco [8], además de JBoss 7.2 y la base de datos Microsoft SQL Server, aunque el requisito de ambos es trivial comparado con los programas previos. El ordenador dispone de 32 GB de RAM, un procesador Intel i7 de décima generación y disco SSD de 500 GB. Cumple con creces los requisitos recomendados para correr los programas sin ningún problema. Ahora, después de haber hecho el trabajo, se confirma que el ordenador es muy potente y que no ha habido problemas en cuanto a capacidad computacional o espacio en la RAM. Los programas cargaban rápido y sin problema.

A.1.2. Microsoft SQL Server

La base de datos que se va a usar es Microsoft SQL Server, elegida por Informatica, [4] la cual recomienda la versión más actualizada.

Para instalar MSSQLSERVER, es preciso dirigirse a la pagina oficial de Microsoft [9] y descargar la versión desarrollador. Posteriormente, se ejecuta el «.exe» y se instala.

Una vez instalado, se crea una nueva instancia de base de datos con la ayuda de la aplicación "SQL Server Installation Center"de Microsoft. Esta también descargará Microsoft Visual C++ de forma automática.

Para proceder con la instalación de MSSQLSERVER hay que habilitar las transacciones distribuidas XA. Desde servicio de componentes se selecciona el ordenador y, a continuación, en las propiedades de "DTC Local", se habilitan las transacciones XA y Acceso DTC. Después se aplican los cambios (A.1).

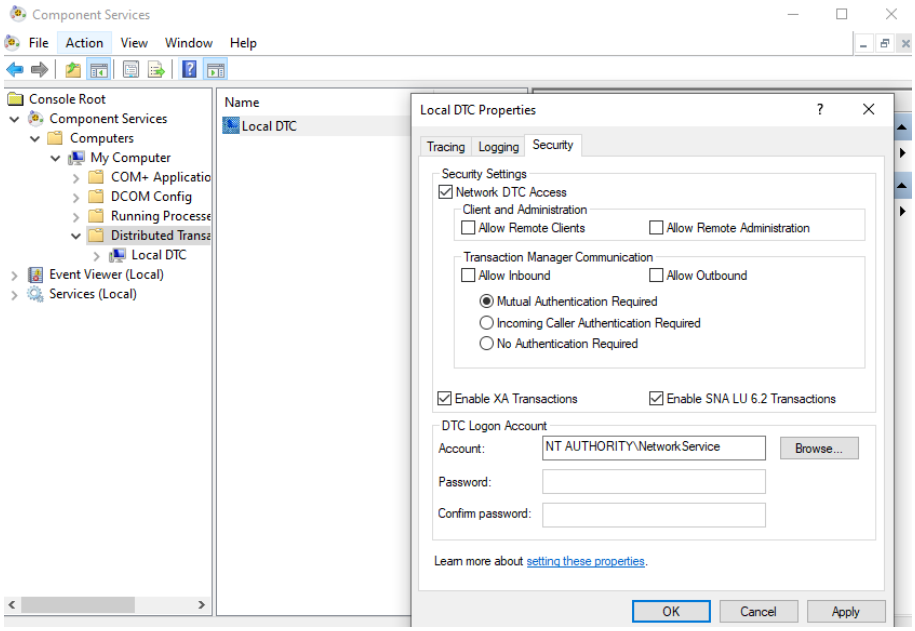

Figura A.1: Panel de configuración de las transacciones distribuidas XA.

También hay que habilitar el tráfico TCP/IP de la máquina local (A.2).

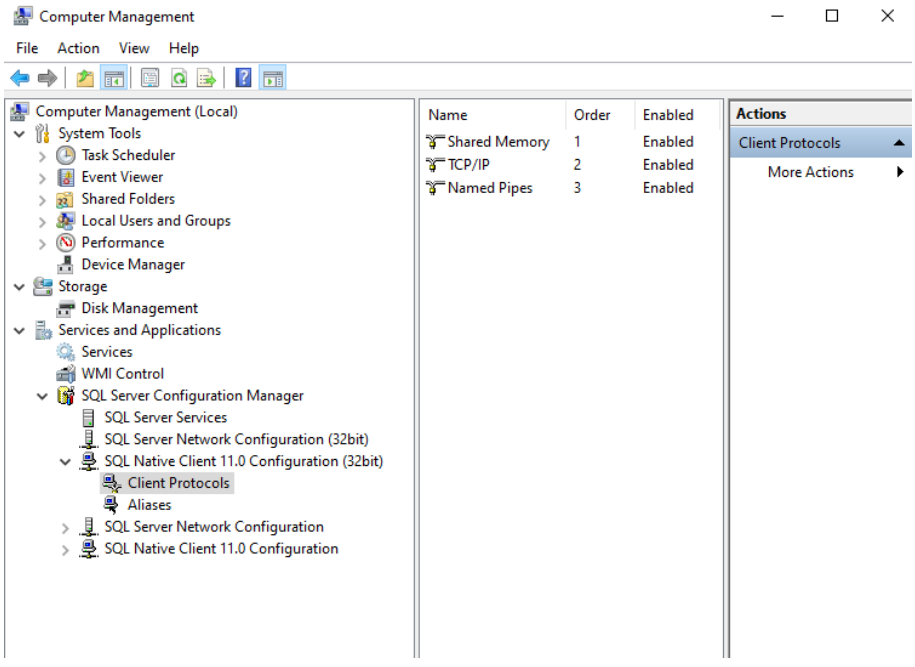

Figura A.2: Ventana de los protocolos de clientes para bases de datos SQL Server.

El siguiente paso es instalar un cliente de base de datos. En este caso se ha escogido «Microsoft SQL Server Management Studio» en su versión 18, pero cualquier otro tipo de cliente de base de datos funcionaría sin problemas.

Ha de comprobarse que el agente de la base de datos esté iniciado y que la instancia esté encendida para ejecutar el *script* de SQL que creará los usuarios, las tablas necesarias y su configuración.

```
--Domain Database
CREATE DATABASE TFG_DOM
GO
USE TFG_DOM
GO
alter database TFG_DOM set READ_COMMITTED_SNAPSHOT ON
GO
create login TFG_DOM with password='TFG_DOM', check_policy = off,
    default_database = TFG_DOM
GO
create user TFG_DOM for login TFG_DOM
GO
exec sp_defaultdb TFG_DOM,TFG_DOM
GO
exec sp_addrolemember db_owner, TFG_DOM
GO
--Domain Object Cache Database
CREATE DATABASE TFG_OBJ
GO
USE TFG_OBJ
GO
alter database TFG_OBJ set READ_COMMITTED_SNAPSHOT ON
GO
create login TFG_OBJ with password='TFG_OBJ', check_policy = off,
    default_database = TFG_OBJ
GO
create user TFG_OBJ for login TFG_OBJ
GO
exec sp_defaultdb TFG_OBJ,TFG_OBJ
GO
exec sp_addrolemember db_owner, TFG_OBJ
GO
--Model Repository
CREATE DATABASE TFG_Model_RS
GO
USE TFG_Model_RS
GO
alter database TFG_Model_RS set READ_COMMITTED_SNAPSHOT ON
GO
create login TFG_Model_RS with password='TFG_Model_RS', check_policy = off,
    default_database = TFG_Model_RS
GO
create user TFG_Model_RS for login TFG_Model_RS
GO
exec sp_defaultdb TFG_Model_RS ,TFG_Model_RS
GO
exec sp_addrolemember db_owner, TFG_Model_RS
GO
--PowerCenter Repository
CREATE DATABASE TFG_PWCR
GO
USE TFG_PWCR
GO
alter database TFG_PWCR set READ_COMMITTED_SNAPSHOT ON
GO
```

```
53 create login TFG_PWCR with password='TFG_PWCR', check_policy = off,
       default_database = TFG_PWCR
54 GO
55 create user TFG_PWCR for login TFG_PWCR
56 GO
57 exec sp_defaultdb TFG_PWCR,TFG_PWCR
58 GO
59 exec sp_addrolemember db_owner, TFG_PWCR
60 GO
61 --Profiling Warehouse
62 CREATE DATABASE TFG_PROFILE
63 GO
64 USE TFG_PROFILE
65 GO
66 alter database TFG_PROFILE set READ_COMMITTED_SNAPSHOT ON
67 GO
68 create login TFG_PROFILE with password='TFG_PROFILE', check_policy = off,
       default_database = TFG_PROFILE
69 GO
70 create user TFG_PROFILE for login TFG_PROFILE
71 GO
72 exec sp_defaultdb TFG_PROFILE,TFG_PROFILE
73 GO
74 exec sp_addrolemember db_owner, TFG_PROFILE
75 GO
76 --Workflow Database
77 CREATE DATABASE TFG_WORKFLOW
78 GO
79 USE TFG_WORKFLOW
80 GO
81 alter database TFG_WORKFLOW set READ_COMMITTED_SNAPSHOT ON
82 GO
83 create login TFG_WORKFLOW with password='TFG_WORKFLOW', check_policy = off,
       default_database = TFG_WORKFLOW
84 GO
85 create user TFG_WORKFLOW for login TFG_WORKFLOW
86 GO
87 exec sp_defaultdb TFG_WORKFLOW,TFG_WORKFLOW
88 GO
89 exec sp_addrolemember db_owner, TFG_WORKFLOW
90 GO
91 --Reference Datawarehouse Repository
92 CREATE DATABASE TFG_REFERDATA_REP
93 GO
94 USE TFG_REFERDATA_REP
95 GO
96 alter database TFG_REFERDATA_REP set READ_COMMITTED_SNAPSHOT ON
97 GO
98 create login TFG_REFERDATA_REP with password='TFG_REFERDATA_REP', check_policy
       = off, default_database = TFG_REFERDATA_REP
99 GO
100 create user TFG_REFERDATA_REP for login TFG_REFERDATA_REP
101 GO
102 exec sp_defaultdb TFG_REFERDATA_REP,TFG_REFERDATA_REP
103 GO
104 exec sp_addrolemember db_owner, TFG_REFERDATA_REP
105 GO
106 --Exception Audit Database
107 CREATE DATABASE TFG_EXCEPAUDIT
108 GO
109 USE TFG_EXCEPAUDIT
110 GO
111 alter database TFG_EXCEPAUDIT set READ_COMMITTED_SNAPSHOT ON
112 GO
```

```
113 create login TFG_EXCEPAUDIT with password='TFG_EXCEPAUDIT', check_policy = off,
        default_database = TFG_EXCEPAUDIT
114 GO
115 create user TFG_EXCEPAUDIT for login TFG_EXCEPAUDIT
116 GO
117 exec sp_defaultdb TFG_EXCEPAUDIT,TFG_EXCEPAUDIT
118 GO
119 exec sp_addrolemember db_owner, TFG_EXCEPAUDIT
120 GO
121 --Reference Data Repository
122 CREATE DATABASE TFG_MON_MRS
123 GO
124 USE TFG_MON_MRS
125 GO
126 alter database TFG_MON_MRS set READ_COMMITTED_SNAPSHOT ON
127 GO
128 create login TFG_MON_MRS with password='TFG_MON_MRS', check_policy = off,
        default_database = TFG_MON_MRS
129 GO
130 create user TFG_MON_MRS for login TFG_MON_MRS
131 GO
132 exec sp_defaultdb TFG_MON_MRS, TFG_MON_MRS
133 GO
134 exec sp_addrolemember db_owner, TFG_MON_MRS
135 GO
```

No todas las bases de datos y usuarios que se crean en el script son utilizados pero serán necesarios en el caso de querer mejorar el producto en un futuro y ampliar el potencial del mismo. Los repositorios necesarios serán, el repositorio de datos y su usuario para la monitorización, el repositorio de flujo de trabajo y finalmente el repositorio que almacenará los servicios de análisis de datos.

Para finalizar la configuración de MSSQLSERVER, Informatica provee un *script* de SQL para habilitar las transacciones XA.

A.1.3. Servidor de Informatica

La empresa Informatica proporciona al cliente las siguientes herramientas (A.3) a través de un ejecutable, además de la licencia del programa.

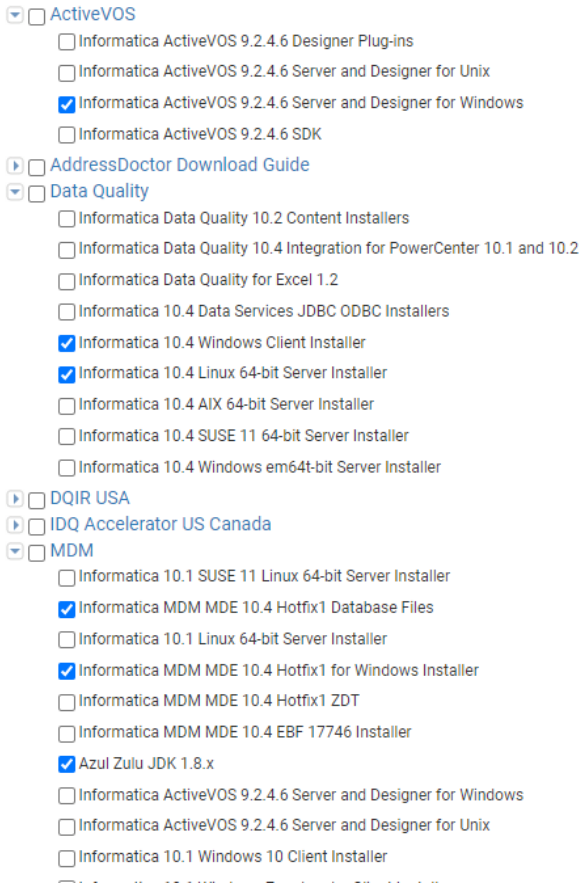

Figura A.3: Interfaz de Informatica donde se descarga el software requerido.

Se empieza por instalar el servidor de Informatica con el ejecutable «install.bat», que se encuentra en la carpeta descomprimida del servidor. Durante el proceso de instalación pedirá un puerto de conexión (mediante el *wizard*) para conectarse mediante HTTPS (por defecto es 8843), así como el puerto del servicio de Informatica (1433). Habrá que abrir una excepción en el *firewall* de Windows para habilitar ambos puertos,
siendo el JDBC de conexión el siguiente:

```
jdbc:informatica:sqlserver://localhost:1433;databaseName=TFG_DOM
```

El usuario y la contraseña son los creados en el *script* de instalación SQL de las bases de datos. En este caso, el usuario y la contraseña son los mismos que el nombre de la base de datos.

Una vez finalizado el proceso, se debe iniciar el servicio Informatica 10.4.0 de Windows y conectarse a través de un navegador a «localhost:8443/administrator», donde se encuentra la consola Administrator de Informatica (A.4).

Dentro de la herramienta Administrator hay que crear todas las diferentes conexiones a las bases de datos.

En la pestaña de conexiones, hay que desplegar el botón de acciones y crear una nueva conexión de tipo SQL Server. Con el usuario y contraseña creados en el *sript* de SQL previamente ejecutado, se añade la cadena de conexión JDBC y se prueba la conexión. Si es correcta, se procede con todas las conexiones restantes.

Una vez creadas todas las conexiones se procede a crear los servicios, que son: el repositorio de datos, la monitorización del repositorio, la herramienta de análisis y el servicio

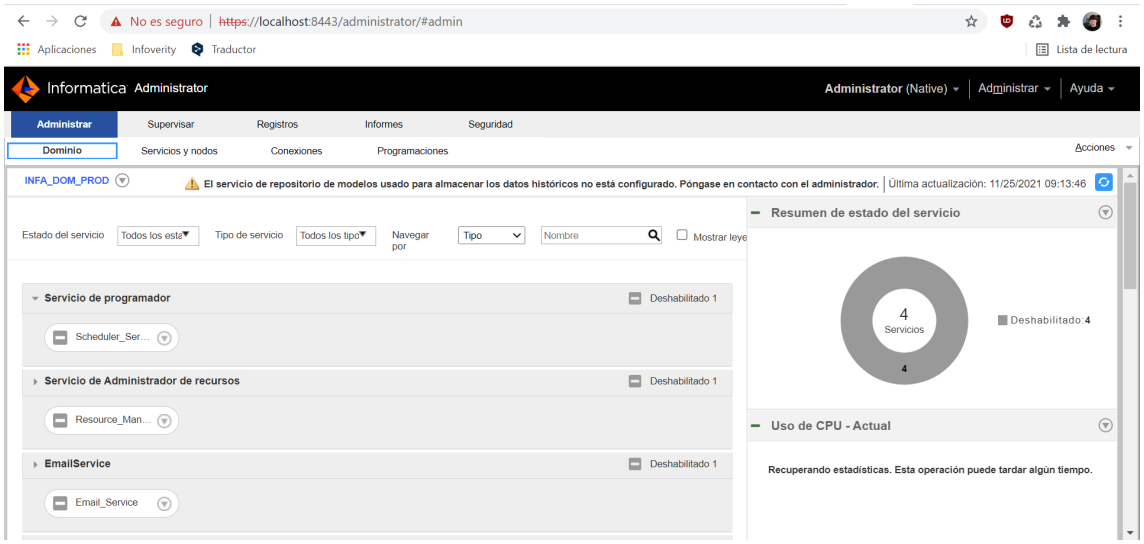

Figura A.4: Interfaz de Informatica Administrator previo a habilitar los servicios.

de integración de datos (A.5). Se conectan todos los repositorios a sus respectivas conexiones. Desplegando el botón de Acciones, se elige el servicio que se desea implementar.

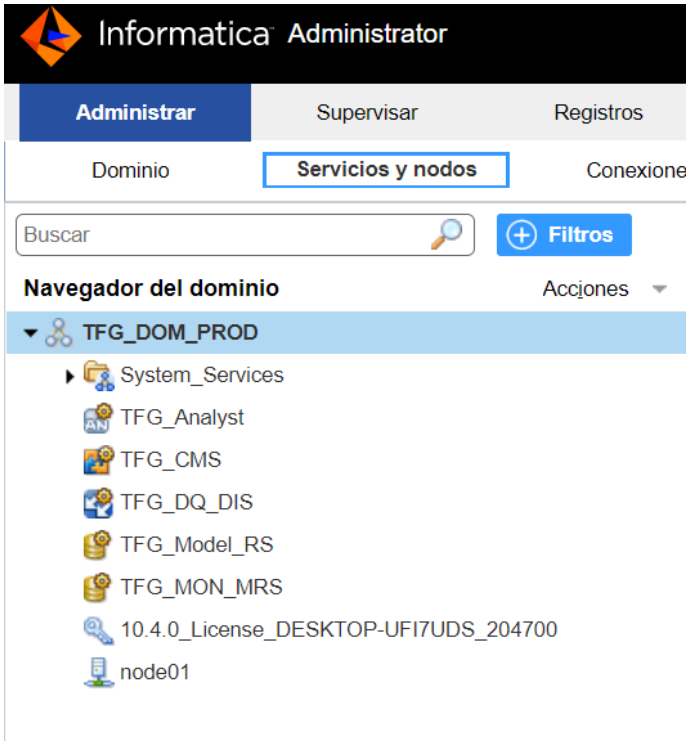

Figura A.5: Interfaz Web que muestra los servicios una vez que han sido creados.

Los servicios están activos y funcionando (A.6).

Continuando con el proceso de configuración del servicio de integración de datos (TFG_DQ_DIS), hay que añadirle un flujo de trabajo (*Workflow*) a la base de datos. Una vez completada la configuración del administrador web, hay que instalar y configurar el cliente de Informatica, que se conectará a los repositorios que se acaban de crear.

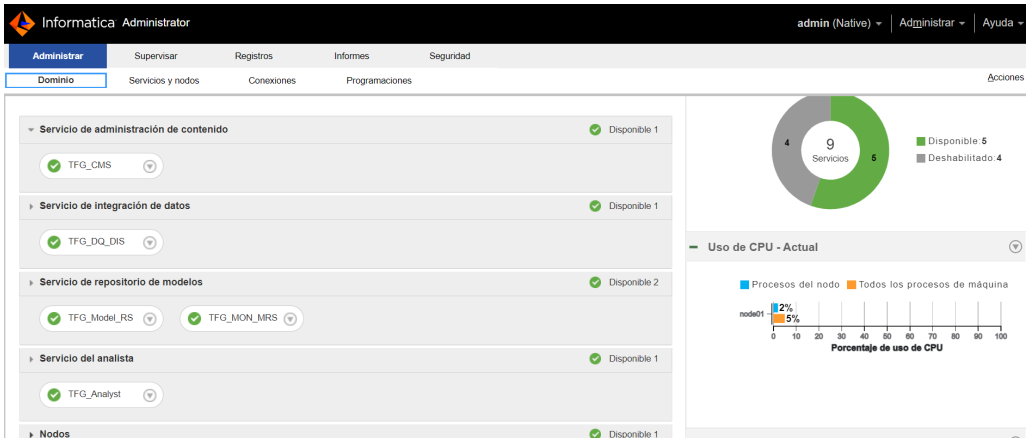

Figura A.6: Administrator con los servicio habilitados.

A.1.4. Cliente de Informatica

En la carpeta donde se han descomprimido los archivos proporcionados por Informatica, se encuentra el «install.bat» del cliente. Con la ayuda del *wizard* de instalación, se deben seguir los pasos e instalar el Developer.

Una vez instalado, debe comprobarse que el servicio de Windows, Informatica 10.4.0, está encendido. Luego, se ejecuta «run.bat» en la carpeta que se acaba de instalar para iniciar el Developer. Cuando esté iniciado, hay que conectarse al repositorio que se ha creado introduciendo el dominio (A.7). Cuando el dominio esté establecido, saldrá la

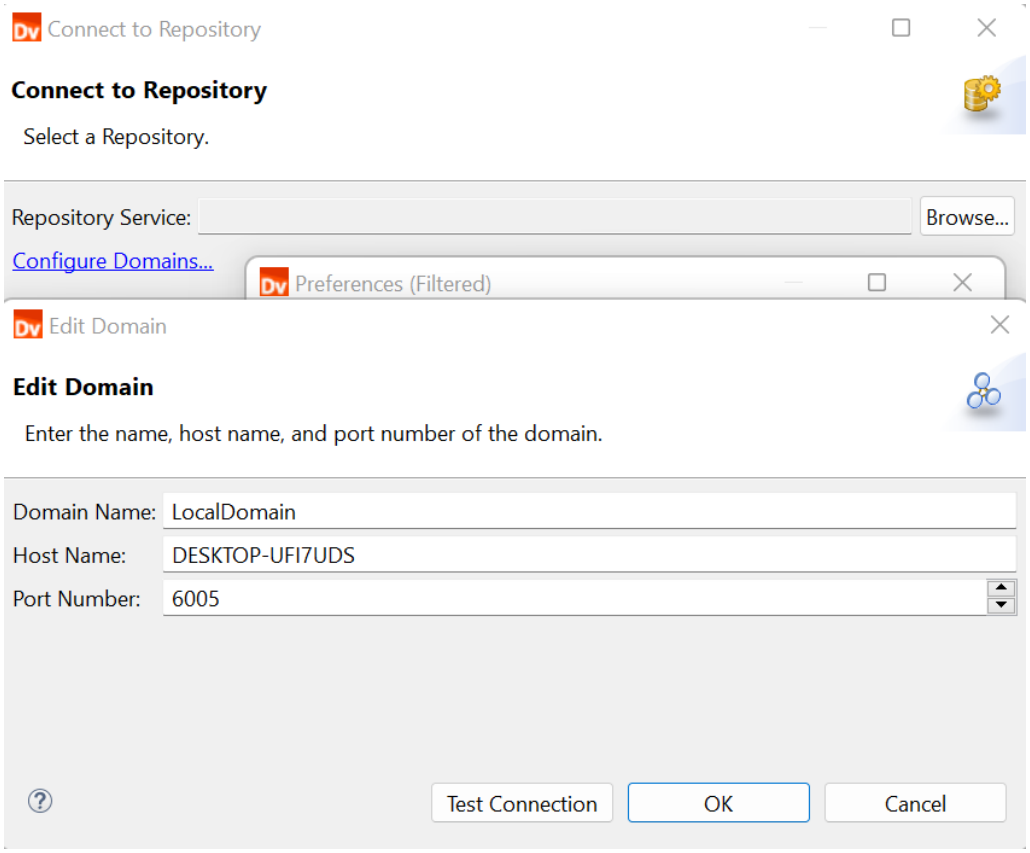

Figura A.7: Creación del dominio *localhost* y el puerto.

estructura de del proyecto, en la cual se debe añadir la carpeta del proyecto TFG (A.8).

Así quedaría finalizada la instalación de Informatica Developer, tanto del cliente como del servidor.

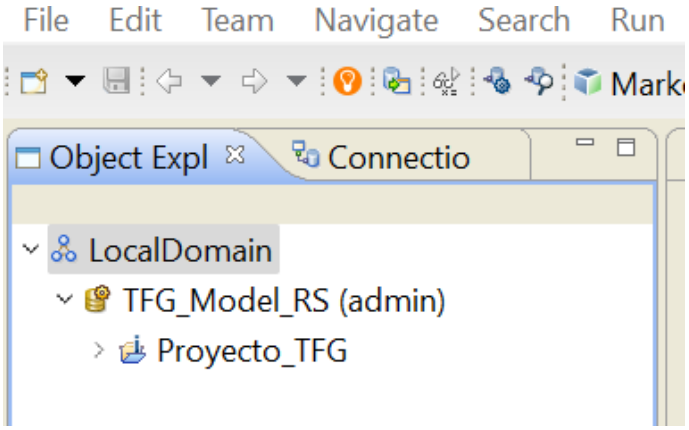

Figura A.8: Muestra del dominio creado con el la carpeta del proyecto TFG.

A.2 Instalación de MDM Hub y el servidor MDM

Una vez instalado Informatica Developer, empieza la segunda fase de la instalación. Esta es MDM, la herramienta encargada de hacer *data governance* y el núcleo de este proyecto.

Al igual que para la instalación de Informatica Developer, la misma empresa Informatica proporciona también en este caso de un instalador con las herramientas que hay que instalar, entre las que se encuentra la versión de Java JDK, así como el archivo de la bases de datos que configura todas las propiedades y atributos para el sistema MDM.

A.2.1. Prerrequisitos

Antes de comenzar con la instalación de la herramienta MDM hay que instalar el servidor JBoss y configurarlo de forma correcta, además de crear las variables de entorno y ejecutar el *script «database»*.

Variables de entorno

Una vez descomprimido el archivo otorgado por Informatica, simplemente hay que ejecutar el instalador de paquetes de Windows del Azul Zulu JDK. Se deben configurar las variables de entorno de Windows para la instalación de MDM, que son:

JAVA_HOME	C:Program Files/Zulu/zulu-8
JAVA_PATH	C:Program Files/Zulu/zulu-8
JRE_HOME	C:Program Files/Zulu/zulu-8/jre

Para comprobar que las variables se han establecido de manera correcta, se ejecuta como administrador el comando «echo %PATH %» y «echo %HOME %» en la consola de comando de Windows.

Archivo Database

Informatica proporciona un *script* de SQL para la configuración de de las bases de datos que usará MDM. Para ejecutarlo, simplemente hay que abrir el cliente de bases de datos y luego abrir el archivo que contiene el *script* y, por último, ejecutarlo.

Servidor Web JBoss

La versión de JBoss recomendada a instalar es la 7.2 [4]. Además de instalar el JBoss, también hay que instalar en *driver* JDBC de Microsoft junto con el servidor.

Para ello, hay que dirigirse a la página de Microsoft y descargar el *driver* 7.2.2 para SQL Server.

También hay que descargar JBoss EAP 7.2. Una vez descargado comenzar la instalación del mismo. Durante la instalación, el programa preguntará si añadir algún *driver*. En este momento, se le indica la ruta del *driver* JDBC previamente descargado. Una vez JBoss esté instalado, se procede a realizar su configuración.

En la ruta de archivos donde está JBoss también se encuentra el archivo «standalone-full.xml». Este archivo contiene la configuración por defecto, junto con el manual de Informatica [4] en la sección «*Configure Server Properties for the Full Profile*». Se deben modelar los atributos para que el servidor y el cliente funcionen de manera correcta.

```
.
.
.
<interfaces>
    <interface name="any">
        <any-address/>
    </interface>
    <interface name="management">
        <inet-address value="${jboss.bind.address.management:127.0.0.1}"/>
    </interface>
    <interface name="public">
        <inet-address value="${jboss.bind.address:127.0.0.1}"/>
    </interface>
    <interface name="private">
        <inet-address value="${jboss.bind.address.private:127.0.0.1}"/>
    </interface>
    <interface name="unsecure">
        <inet-address value="${jboss.bind.address.unsecure:127.0.0.1}"/>
    </interface>
</interfaces>
.
.
.
```

Se muestra el uso de la dirección IP 127.0.0.1 porque la máquina a la que se conecta es local.

Otro fichero que se debe configurar es el ejecutable de Windows «standalone-conf.bat», donde se detallan las rutas de los logs, puerto y host de MDM, entre otros.

```
.
.
.
set "JAVA_OPTS=%JAVA_OPTS% -De360.connection.channel=HTTP   "
set "JAVA_OPTS=%JAVA_OPTS% -De360.mdm.port=8080"
set "JAVA_OPTS=%JAVA_OPTS% -De360.mdm.host=localhost"
set "JAVA_OPTS=%JAVA_OPTS% -Dorg.jboss.boot.log.file=C:\Informatica\JBoss\
    standalone\log\boot.log   "
```

```
set "JAVA_OPTS=%JAVA_OPTS% -Dlogging.configuration=file:C;\Informatica\JBoss\
    standalone\configuration\logging.properties "
set "JAVA_OPTS=%JAVA_OPTS% -Djboss.home.dir=C;\Informatica\JBoss "
.
.
.
```

Se debe ejecutar JBoss mediante el siguiente comando, que se ejecuta adquiriendo la configuración del archivo «standalone-full.xml»:

```
standalone -c standalone-full.xml -b 0.0.0.0 -bmanagement 0.0.0.0 -Djboss.as.
    management.blocking.timeout=5000
```

Para comprobar que JBoss funciona, hay que ir al propio JBoss y entrar con las credenciales creadas en el proceso de instalación:

http://localhost:9990/console/index.html

ActiveVOS

Aquí es donde se crea el usuario del ActiveVOS.

Primero, hay que dirigirse a la carpeta «bin» de JBoss donde se encuentra add-user.bat y ejecutarlo. Para crear un usuario de ActiveVOS se deben seguir las siguientes instrucciones.

Type of user	'Application User'
Username	'avos'
Password	'avos'
Confirm password	'avos'
Group	'abAdmin'
Is the configuration correct?	'Y'

A.2.2. Instalación de MDM

A continuación, hay que ejecutar el hub_installa.exe que proporciona Informatica. Se sigue el *wizard* de instalación y ante la pregunta de si se desea instalar ActiveVOS, hay que señalar que sí.

* Mientras se ejecuta la instalación se pueden ver en la carpeta /hub/server/logs/-PostInstallSetup.log los documentos que está creando el instalador y comprobar si aparece algún error.

Postinstalación del HUB Server

Para llevar a cabo la última fase de la instalación de las herramientas necesarias para la realización del presente trabajo, se deben seguir los pasos a continuación listados.

- Ejecutar el comando, en C:/*ruta MDM*/hub/server.

```
postInstallSetup.bat - Ddatabase.password=avos -Davos.username=avos -
    Davos.password=avos -Davos.jdbc.database.password=avos -Davos.jdbc.
    database.username=avos
```

- Crear *system* y *ors* [10].
  ejecutar ant.bat create_system
  ejecutar ant.bat create_ors
- Importar *system* y *ors* [11].
  ejecutar ant.bat import_system
  ejecutar ant.bat import_ors
- Ejecutar:

```
java -classpath siperian-api.jar;siperian-common.jar;siperian-server.jar
com.delos.util.PublicKeyBasedEncryptionHelper admin
```

- Comprobar que ActiveVOS funciona.

```
http://localhost:8080/activevos/activevos_dashboard_landing.action
```

- Ir a:

```
localhoat:8080/cmx.
```

- Descargar el ejecutable de Informatica: MDM Hub.
- Ejecutar MDM Hub.

Una vez realizados estos pasos, ya estaría finalizada la instalación.