Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

Implementación de un sistema de información con tecnología MDM

Trabajo Fin de Grado

Grado en Ingeniería Informática

Autor: Héctor Ramiro Serrano

Tutor: Ignacio Gil Pechuán

2014 - 2015

Resumen

La administración de datos maestros o críticos en una organización para su tratamiento y procesamiento, con el fin de obtener el conocimiento necesario para la toma correcta de decisiones, es vital para la estrategia global y la mejora en la competitividad de dicha organización.

Esta tarea se vuelve especialmente compleja cuando la información crítica proviene de diferentes fuentes de datos, pertenecientes a diferentes ámbitos de negocio.

El presente TFG tiene como objetivo construir un sistema información capaz de fusionar datos maestros de múltiples fuentes sobre un modelo único y consolidado, utilizando para ello tecnología Master Data Management, y que permita a una empresa determinada implantar un sistema de Business Intelligence que sirva de apoyo a la toma de decisiones.

Palabras clave: sistema de información, Master Data Management, Business Intelligence, integración, datos maestros, datos críticos.

Abstract

La administración de datos maestros o críticos en una organización para su tratamiento y procesamiento, con el fin de obtener el conocimiento necesario para la toma correcta de decisiones, es vital para la estrategia global y la mejora en la competitividad de dicha organización.

Esta tarea se vuelve especialmente compleja cuando la información crítica proviene de diferentes fuentes de datos, pertenecientes a diferentes ámbitos de negocio.

El presente TFG tiene como objetivo construir un sistema información capaz de fusionar datos maestros de múltiples fuentes sobre un modelo único y consolidado, utilizando para ello tecnología Master Data Management, y que permita a una empresa determinada implantar un sistema de Business Intelligence que sirva de apoyo a la toma de decisiones.

Keywords: information system, Master Data Management, Business Intelligence, integration, master data, critical data.

Tabla de contenidos

1. Introducción 11
   1.1 Objetivos 12
      1.1.1 Objetivos generales 12
      1.1.2 Objetivos específicos 12
   1.2 Justificación 13
   1.3 Estructura de la memoria 15
2. Fundamentos teóricos 16
   2.1 Business Intelligence 16
      2.1.1 Definición 16
      2.1.2 Utilidades 16
      2.1.3 Repercusión en la organización 17
      2.1.4 Éxito de implantación 18
   2.2 Cuadro De Mando 18
      2.2.1 Indicadores 19
   2.3 Master Data Management 21
      2.3.1 Definición 21
      2.3.2 Repercusión 23
      2.3.3 Éxito de implantación 23
      2.3.4 Soluciones similares 24
3. Caso de estudio 26
   3.1 Sistemas involucrados 26
      3.1.1 Supply Chain Manager 26
      3.1.2 Enterprise Resource Planning 26
      3.1.3 Computerized Maintenance System 26
      3.1.4 Geographic Information System 27
      3.1.5 Content Management System 27
   3.2 Flujo de trabajo inicial 27
   3.3 Funcionamiento tras la implantación 29
4. Metodología 31
   4.1 Arquitectura Cliente-Servidor 31
      4.1.1 Definición 31

4.1.2 Componentes y funcionamiento 32
4.1.3 Características 33
4.1.4 Justificación 33
4.2 Softwares de Virtualización 33
4.2.1 VMware Player 34
4.2.2 Oracle VirtualBox 34
4.2.3 Otras alternativas 35
4.3 Interconexión de redes 35
4.3.1 Adaptador Puente 35
4.3.2 NAT 35
4.4 Sistemas operativos 36
4.4.1 CentOS 6 36
4.4.2 Windows Server 2008 R2 Enterprise 36
4.4.3 Windows 8.1 37
4.4.4 Windows 10 37
4.5 Metodologías Ágiles 37
4.6 Lenguajes de programación 38
4.6.1 Microsoft .NET C# 38
4.6.2 Visual Basic 38
4.6.3 SQL 38
4.7 Sistemas de gestión de bases de datos relacionales 39
4.7.1 Oracle Database 39
4.7.2 Microsoft Access 39
4.7.3 Microsoft SQL Server 39
4.8 Software de desarrollo 40
4.8.1 Visual Studio 40
4.8.2 Microsoft Access 40
4.9 MDM 40
4.9.1 Informatica MDM 41
4.9.2 Plataforma 42
4.9.3 Target Data Model 43
4.9.4 Tablas Landing 43
4.9.5 Mappings 44
4.9.6 Tablas Staging 44
4.9.7 Batch Groups 45
4.9.8 Actualización por Last Update Date 45

4.10 Ilustración 46
4.10.1 yEd 46
4.10.2 UML 46
4.11 Otras herramientas utilizadas 46
4.11.1 WinMerge 46
4.11.2 Notepad++ 46
4.11.3 SoapUI 47
4.11.4 Agent Ransack 47
5. Implementación 48
5.1 Definición de la arquitectura 48
5.2 Sistema MDM 50
5.2.1 Creación de un Operational Record Store 50
5.2.2 Diseño del Tarjet Data Model 51
5.2.3 Implementación de los Base Objects 58
5.2.4 Definición de Landings 59
5.2.5 Listas de valores 62
5.2.6 Definición de Staggings 63
5.2.7 Definición de Mappings 65
5.2.8 Creación de los Batch Groups 75
5.3 Simulación de las fuentes de datos 76
5.3.1 Sistema ERP 76
5.3.2 Sistema SCM 78
5.3.3 Sistema CMMS 79
5.3.4 Sistema GIS 80
5.3.5 Sistema CMS 81
5.3.6 Método de carga 82
5.4 Prueba unitaria 82
5.4.1 Definición de indicadores 82
5.4.2 Interfaz gráfica 83
5.5 Ejemplos de uso de MDM 84
6. Conclusiones 85
6.1 Retos y problemas 85
6.2 Líneas de trabajo abiertas 86
Anexo A: Bibliografía 88
Anexo B: Ejemplos de uso de MDM 89
B.1 Mapeo de nuevos campos 89

B.1.1 Tipo: Change Request 89
B.1.2 Análisis 89
B.2 Listas de valores suministradas por fuentes 90
B.2.1 Tipo: Test 90
B.2.2 Análisis 90
B.2.3 IMPACTO 91
B.3 Incremento de la precisión de un atributo 91
B.3.1 Tipo: Change request 91
B.3.2 Impacto 91
B.3.3 Impacto 92
B.3.4 Solución 93
B.4 El resultado del mapeo de PKEY_SRC_OBJECT es nulo 94
B.4.1 Tipo: Error 94
B.4.2 Análisis 94
B.5 Fecha nula o perteneciente al futuro 94
B.5.1 Tipo: Error 94
B.5.2 Análisis 95
B.6 Clave ajena con valor nulo 95
B.6.1 Tipo: Error 95
B.6.2 Análisis 95
B.6.3 Impacto y solución 96
B.7 Registros duplicados en las tablas Landing 97
B.7.1 Tipo: Error 97
B.7.2 Análisis 97
B.8 No se encuentra el valor de la Lookup 98
B.8.1 Tipo: Error 98
B.8.2 Análisis 98
B.9 Actualización por LUD 99
B.9.1 Tipo: Test 99
B.9.2 Análisis 99
B.9.3 Impacto 100
B.10 Allow Null Update 100
B.10.1 Tipo: Test 100
B.10.2 Análisis 100
B.11 Apply Null Values 101
B.11.1 Tipo: Test 101

B.11.2 Análisis 101
B.12 Trust 103
B.12.1 Tipo: Test 103
B.12.2 Análisis 103
B.12.3 Impacto 104

1. Introducción

Actualmente, existe en las organizaciones la necesidad de establecer mediciones y controles fiables sobre sus datos maestros en aquellos aspectos o áreas críticas y relevantes la empresa, con el objetivo de conocer bien su funcionamiento y poder tomar las decisiones estratégicas y de negocio correctas. El presente Trabajo de Fin de Grado describe y analiza estas necesidades, a la vez que se expone un caso práctico de estudio, en el que una empresa nos pide un Sistema de información capaz de interpretar de forma lo más fidedigna y precisa posible la realidad de su organización, teniendo en cuenta que sus datos están distribuidos y repetidos en 5 sistemas distintos y fuertemente desacoplados. Concretamente, los datos que se manejan son los relativos al ciclo de vida de los activos (entendiendo como activo un recurso físico propiedad de la empresa), desde que se solicita su compra, hasta que hay que tomar la decisión de desmantelarlo, incluyendo todo el proceso de planificación del mantenimiento. Mediante un sistema de Inteligencia de Negocio o Business Intellience (de ahora en adelante BI), dicha empresa quiere realizar todas las tareas de toma de decisiones y poder realizar la planificación y control del ciclo de vida completo de sus activos.

Mediante la utilización de la herramienta Informatica MDM, basada en tecnología Master Data Management (de ahora en adelante MDM), se implementará un sistema de integración y consolidación de datos maestros, en el que cada sistema informático de la organización aporta sus datos para que este proceda a procesar la información para insertarla e integrarla dentro de una base de datos maestra. Una vez que se hayan consolidado los datos, el sistema BI explotará toda la información consolidada y podrá dar apoyo para tomar las mejores decisiones de negocio.

La realización del presente trabajo tuvo lugar dentro de un contexto empresarial, de la mano de la compañía Arhis Competency Center, dedicada a dar soluciones de integración y consolidación de datos de grandes empresas y que ha dotado al proyecto de una perspectiva centrada en la realidad de la empresa, enfocando la resolución del problema hacia la satisfacción del cliente.

1.1 Objetivos

Los objetivos del trabajo realizado se han dividido en 2 grupos: objetivos generales y objetivos específicos.

Los objetivos generales están constituidos por aquellos conocimientos, métodos, herramientas, tecnologías, etc. Derivados del desarrollo del proyecto. Los objetivos específicos son aquellos que competen exclusivamente a la solución del problema propuesto.

1.1.1 Objetivos generales

Los objetivos generales se listan a continuación:

- Exponer la importancia de manejo correcto de los datos maestros de la organización y la extracción de conocimiento de los mismos.
- Exponer los beneficios que eso conllevaría.
- Enfocar el problema desde el punto de vista del cliente, para entender las necesidades y problemas que pueda haber en casos reales.
- Simular un entorno de interacción y manejo de datos donde intervienen varios sistemas débilmente acoplados.
- Describir los problemas y la complejidad que conlleva la integración de varios sistemas débilmente acoplados en un único sistema consolidado.
- Exponer y describir detalladamente la solución propuesta (Informatica MDM), junto con la exposición de posibles alternativas para su comparación
- Consolidar conocimientos en bases de datos.
- Consolidar conocimientos en sistemas de información estratégicos.
- Adquirir las habilidades para manejar y usar las herramientas MDM, exponiendo casos como cambios en los requerimientos del cliente, errores y problemas que podrían surgir o configuraciones distintas a un mismo problema o planteamiento.

1.1.2 Objetivos específicos

Los objetivos específicos están relacionados con el desarrollo del caso de estudio propuesto, y se listan a continuación:

- Identificar las necesidades reales del cliente en base a sus requerimientos.
- Definir los indicadores en base a los datos que se consideren críticos o maestros y definir los indicadores.
- Planear y diseñar la unificación de las distintas bases de datos

- Desarrollar un sistema BI en forma de cuadro de mando que permita validar el sistema MDM, para su integración con un sistema BI profesional externo que el cliente quiera poner en marcha después de la consolidación de los datos.
- Solucionar el problema de la unión y consolidación de los datos provenientes de distintas fuentes mediante tecnología MDM para desarrollar un punto único de referencia al que pueda consultar el cuadro de mando.
- Ofrecer la Mejor Versión de la Verdad o Best Version of the Truth (de ahora en adelante BVT) de los datos maestros de la empresa.

1.2 Justificación

La gobernabilidad de los datos es un concepto que ha ido ganando fuerza en los últimos años. Es indiscutible que, si una empresa quiere lidiar en igualdad de condiciones con sus competidores, además de llevar un buen control de los datos de la empresa, debe de saber cómo interpretarlos correctamente.

Ya en 1987, la Comisión de Productividad Industrial del Instituto de Tecnología de Massachusetts instó a que los sectores industriales creasen técnicas para medir y aumentar la eficacia de los sectores de producción. Estas son algunas de las repercusiones que se desprenden de estas prácticas:

- Mejores datos para dar apoyo a la toma de decisiones
- Mejora de operaciones a través de análisis
- Mejora la posición competitiva
- Mejorar la estrategia
- Mejorar la capacidad de predicción de futuras acciones

La necesidad de trabajar con la mejor versión de la información se convierte, pues, en vital, ya que se trata de operaciones de gran sensibilidad para la empresa, en la que el margen de error debe acortarse lo máximo posible. En los enfoques clásicos, el problema es que suelen manejar datos pertenecientes al pasado (por lo que podrían estar obsoletos) cuya utilidad es relativa, y que puede que no haya una relación entre los indicadores elegidos y, por tanto, no tengan un enfoque integrador. Otros errores que pueden cometerse son, por ejemplo, medir demasiadas cosas, dedicar demasiado tiempo a la medición o medir las cosas con valores equivocados. Cometer cualquiera de estos errores tendría como consecuencia la elevación de costes.

Robert Eccles dijo en un artículo llamado “The performance measurement manifesto” (‘Manifiesto sobre la medida del rendimiento’), publicado en la revista Harvard Business Review en 1991, que hay cinco áreas de actividad en las que habría que incidir en una empresa:

1. Desarrollar una arquitectura de la información que permita identificar los datos necesarios para cumplir la estrategia, pensar cómo se generan estos datos y establecer unas normas que regulen el flujo de información.
2. Instalar una tecnología que soporte esta estructura.
3. Adecuar unos incentivos al nuevo sistema.
4. Utilizar recursos económicos externos a la empresa.
5. Diseñar un proceso para que los cuatro puntos anteriores se cumplan.

Por eso, y sobre todo cuando se tienen varias fuentes con versiones y visiones distintas de los datos de la empresa, se hace patente la necesidad de una tecnología capaz de ofrecer seguridad con los datos con los que se está trabajando.

En el caso de estudio propuesto (el seguimiento del ciclo de vida de los activos de la empresa), se exponen todas estas tesituras de una forma práctica, ilustrando con un cuadro de mando, los beneficios de disponer un sistema MDM que integre y cohesione los datos.

Los activos en una empresa son determinantes a la hora de conseguir los objetivos del negocio. Su disponibilidad, la reducción de fallos, su correcto funcionamiento, una correcta planificación del mantenimiento, etc., tiene un impacto directo sobre los resultados de la organización. La capacidad de responder con rapidez, calidad y consistencia a los cambios y requerimientos del negocio está ligada con su éxito. Con esto, se consigue:

- Mejorar los servicios
- Disminuir los costos
- Mejorar la gestión de los riesgos

En definitiva, maximizar el rendimiento y el valor a lo largo del ciclo de vida de los activos de la empresa alineándolos con los objetivos del negocio y la estrategia empresarial, con lo que se conseguiría mayores beneficios por la correcta utilización de los activos y la reducción de costes en compras, cotes de inventario, costes de material, costes de mantenimiento, etc.

1.3 Estructura de la memoria

En este apartado se exponen los capítulos principales en orden, junto con una breve descripción de los temas tratados en cada uno:

- *Fundamentos teóricos*: Se explica los conceptos en los que se fundamenta del trabajo: BI y MDM. Se exponen soluciones alternativas, así como la razón de su utilización y los beneficios que aporta la combinación de las técnicas utilizadas.

- *Caso de estudio*: Se describe la definición del caso de estudio.

- *Metodología*: Se exponen las herramientas, tecnologías, métodos y procedimientos con los que se ha trabajado en algún momento y con los que, finalmente, se ha llevado a cabo la solución al problema planteado, y que sirve como ejemplo para ilustrar la solución a proyectos reales, recorriendo las etapas recorridas a lo largo de todo el trabajo realizado hasta la solución final.

- *Implementación*: Se muestra el diseño en base a los requerimientos del caso de estudio, se describe el proceso de desarrollo paso a paso y, finalmente, se describe su implantación.

- *Ejemplos de uso en MDM*: Se hace una introducción al manejo y mantenimiento del sistema MDM una vez diseñado e implantado, con casos que podrían darse ya sea por requerimientos de cambios del cliente, por fallos del sistema o por errores de diseño.

- *Conclusiones*: Se realiza la valoración general del trabajo realizado, los resultados obtenidos, las metas alcanzadas, las limitaciones y los posibles caminos a seguir en caso de una ampliación futura.

- *Anexo A*: Bibliografía.

- *Anexo B*: Ejemplos de uso de MDM.

2. Fundamentos teóricos

2.1 Business Intelligence

El concepto de Business Intelligence (de ahora en adelante BI) hace referencia a un concepto que ha adquirido especial relevancia en las últimas décadas, por sus implicaciones tanto a nivel productivo como estratégico para las organizaciones.

2.1.1 Definición

Hans Peter Luhn, investigador de IBM, definió en 1958 el BI como "La capacidad de comprender las interrelaciones de los hechos presentados en tal forma como para orientar la acción hacia una meta deseada". Desde entonces, el concepto ha ido evolucionando. En 1989, Howard Dresner, analista de Gartner Group, lo describiría como "los conceptos y métodos para mejorar la toma de decisiones empresariales mediante el uso de sistemas basados en hechos de apoyo". En otras palabras, es la habilidad de extraer información de los datos y transformarla en conocimiento, de tal forma que se pueda optimizar el proceso de toma de decisiones.

En la actualidad, el concepto de BI incluye una amplia categoría de metodologías, aplicaciones y tecnologías que permiten reunir, acceder, transformar y analizar los datos, transacciones e información no estructurada (interna y externa) y que proporciona información privilegiada para responder a los problemas del negocio.

2.1.2 Utilidades

Un BI puede realizar tareas sencillas como informes (predefinidos, a la medida, consultas, etc.) o más orientadas al análisis (estadísticas, pronósticos, Data Mining, etc.):

- Sistemas de informes: o de Reporting, están formados por un generador de informes y por una herramienta de consulta para el usuario. El tipo de información generado es estático.
- Análisis OLAP: Analiza la información de forma dinámica desde diferentes dimensiones o puntos de vista.
- Cuadros de mando: o Dashboards, son paneles de control que permiten hacer el seguimiento de la actividad empresarial en relación con los objetivos de forma simple y gráfica. Esta es la utilidad que se ha desarrollado en el presente trabajo, descrita en detalle en el punto 2.2.

- Minería de datos: o Data Mining, permiten analizar los datos para descubrir patrones de conducta y poder hacer previsiones y predicciones.

En cualquier caso, deberá garantizar en todo momento los siguientes aspectos:

- Acceso de los usuarios a la información con independencia de la procedencia de los datos.
- Proveer herramientas de análisis que apoyen la toma de decisiones.
- Ofrecer una solución al alcance de cualquier usuario con independencia de su nivel técnico.

Existen herramientas como Balanced Scorecard (BSC), Decision Support System (DSS), Execiutive Information System (EIS), etc. La fuente que provee los datos a un sistema BI podría ser un Datamart, un Data Warehouse, etc.

A partir de este momento, cuando hablemos de sistema BI nos referiremos a un software o aplicación que servirá como sistema de información al que se le aportarán una serie de datos extraídos de diferentes fuentes y del cual extraeremos el conocimiento necesario para dar apoyo a la toma de decisiones.

2.1.3 Repercusión en la organización

Una adecuada gestión del conocimiento se hace necesaria si se quiere garantizar el éxito dentro de un mercado competitivo. La utilización de la información para generar conocimiento produce mejoras en los procesos de negocio y guían a las organizaciones a tener operaciones mucho más efectivas y optimizadas.

Algunas ventajas que se derivan de contar con un esquema de BI en la organización son las siguientes:

- Disposición de la información correcta en el momento adecuado. Desaparece la necesidad de consultar la información a diferentes departamentos o sistemas. La información está en un punto de referencia común y se puede extraer de manera sencilla y en tiempo real.
- Provee la capacidad para evaluar distintos escenarios al mismo tiempo, con lo que se pueden analizar diferentes situaciones que podrían afectar al negocio.
- Se pueden definir indicadores que permitan medir el desempeño del negocio.
- Agrupa información de distintas áreas.
- Genera capacidad de reacción a decisiones imprevistas con un nivel de riesgo menor, esto es producto del análisis de diferentes escenarios.

- Capacidad de retroalimentar el conocimiento adquirido, ya que se mantienen disponibles las decisiones tomadas y el impacto que generó.

2.1.4 Éxito de implantación

Cabe la posibilidad que un proyecto de BI no concluya con éxito o no cumpla con las expectativas. Algunos factores que podrían influir son los siguientes:

- Datawarhouse no acorde con las necesidades reales.
- Presupuesto y calendario ajustados.
- Datos de origen erróneos, sobre todo si se dispone de varias fuentes. Duplicidades, errores, caracteres erróneos (implican un proceso de ETL más costoso, mayor tamaño de Base de datos y peor rendimiento).
- Mala elección de los consultores o excesiva rotación entre ellos.
- No contar con apoyo ni impulso de la dirección general.
- No considerar el diseño como un factor relevante.

Una de las claves para poder desarrollar un proyecto de BI con éxito es seleccionar los indicadores y la herramienta más adecuada para implantar el sistema. La selección de estos factores debe estar en función de aspectos como:

- La información que se necesita.
- Con qué propósito se quiere la información.
- A quién va dirigida.
- Aspectos técnicos.

2.2 Cuadro De Mando

James Harrington cuenta que en una planta de IBM (donde estuvo trabajando 40 años) en Havant (Inglaterra), cuando se empezó a proporcionar retroalimentación a cada operario de ensamblaje, el nivel de defectos disminuyó en un 90%. Por eso, la retroalimentación debe ser relevante e inmediata.

El cuadro de mando se encarga de organizar un sistema de indicadores definido y permite que se visualice de forma gráfica y clara toda esa información, proporcionando así la retroalimentación necesaria e informando de la evolución de las características del negocio que se hayan considerado relevantes.

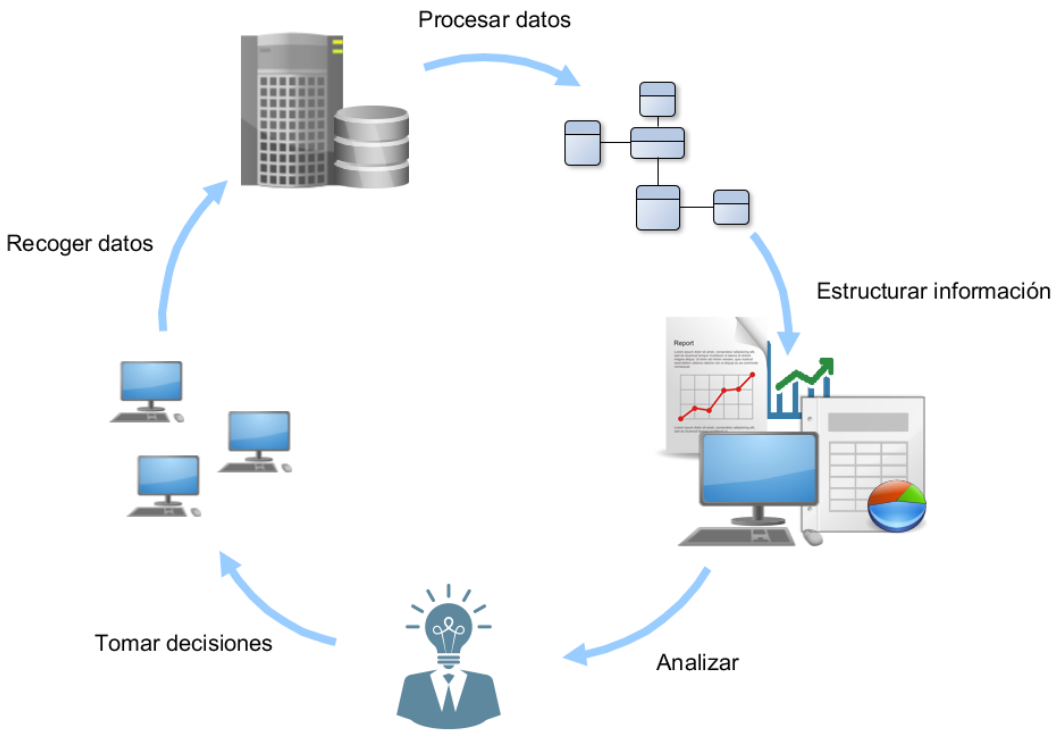


fig. 1 Flujo de información con un cuadro de mando

2.2.1 Indicadores

Los indicadores nacen de la idea de que se puede mejorar aquello que se puede medir. Se puede decir que las mediciones forman parte de nuestra vida (nada más nacer, ya nos están midiendo y pesando).

Citando a Lord Kelvin: "Cuando puedes medir aquello de lo que estás hablando y expresarlo en números, puede decirse que sabes algo acerca de ello. En caso contrario, tu conocimiento es muy deficiente y poco satisfactorio".

Lo que marcará la dirección a seguir serán los indicadores correctamente interpretados. En cuanto al número de indicadores, Price Waterhouse expuso en 1997 que "si se tienen demasiadas medidas, es como si no se tuviera ninguna, 5 o 6 para cada proceso importante".

Los indicadores deben poder identificarse fácilmente (no debe haber dificultad en medirlos), deben referirse a algo representativo de la mejora buscada, debe ser fácilmente comprensible, y deben formar parte de un conjunto, sin dar más importancia a unos sobre otros.

Hay tres tipos de indicadores:

- Preindicadores, que se identifican antes de que ocurran los hechos.
- Indicadores concurrentes, que se establecen inicialmente por adelantado pero evolucionan a lo largo del tiempo.
- Indicadores terminales, que se realizan al terminar los hechos.

Algunos de los indicadores más utilizados son:

2.2.1.1 Consumo de recursos

La eficiencia se puede medir en función de la disminución de recursos (si disminuimos los recursos, aumentamos la productividad). Por ello, los recursos consumidos es uno de los indicadores más usados.

Suelen medirse costos, tiempos empleados, número de personas, plazos de tiempo, materiales usados, desgaste de máquinas, etc.

2.2.1.2 Expectativas frente a realidad

El comparar los resultados reales con los previstos, y medir la desviación (positiva o negativa), es en sí un indicador, como pueden ser el porcentaje de cumplimiento real o el porcentaje de desviación.

2.2.1.3 Ratios de gestión

Son el resultado de la comparación de dos cifras significativas que nos permiten seguir el comportamiento y controlar cualquier área de la organización. Un ejemplo muy sencillo de radio sería la velocidad de un coche, la cual medimos en kilómetros por hora.

Hay que destacar que los ratios únicamente nos indican lo que está ocurriendo, somos las personas los que debemos interpretar y tomar las decisiones.

Los ratios pueden interpretarse de varias formas: en valor absoluto (si se tiene un criterio preestablecido), comparando con datos históricos, comparando con lo previsto, comparando con la competencia o comparando internamente.

2.2.1.4 Otros

Hay otros indicadores que se podrían tener en cuenta como el número de errores, porcentaje de piezas defectuosas, procedimientos cumplidos, etc.

2.3 Master Data Management

La tecnología Master Data Management (de ahora en adelante MDM) es una solución que ayuda a las organizaciones a mejorar la productividad e impulsar el rendimiento operativo, mejorando la precisión de la información y el intercambio de datos dentro y fuera de la empresa.

2.3.1 Definición

MDM es una disciplina empresarial, cada vez más popular, que está diseñada para eliminar el elevado número de errores, redundancias e inconsistencias que existen en los diversos y fragmentados entornos de información actuales. Su principal objetivo es crear un punto de referencia único, preciso y consistente para todos los elementos comunes de datos a lo largo de los diversos sistemas de información y orígenes de datos que una empresa mantiene, así como ofrecer lo que se conoce como BVT (Best Version of the Truth o Mejor Version de la Vaerdad).

Cuando se habla de Master Data (datos maestros) se está haciendo referencia a la información de mayor valor en una organización. Representan información central sobre el negocio y las relaciones entre ellos. En cualquier organización existe un conjunto de datos central, utilizado a través de muchos procesos, unidades de negocio, aplicaciones operacionales y de soporte a decisiones. En otras palabras, los datos maestros son los que definen a la organización.

Cabe destacar que el almacenamiento de estos datos no es algo nuevo, ya que ha ocurrido desde el inicio del procesamiento electrónico de datos, pero la mayor parte de las aplicaciones y bases de datos construidas para administrar la información de clientes y productos se diseñaron para soportar una línea de negocio en particular, en vez de la organización en su conjunto. El problema es que cuando una organización crece, se crean nuevas aplicaciones para el manejo de información de nuevas líneas de negocio, haciendo que cada una de estas aplicaciones contenga su propia copia de los datos maestros, es decir, que se encuentren de forma redundante (y en la mayoría de ocasiones inconsistente) en múltiples ‘silos’. Al contar con múltiples versiones de los datos maestros, la implementación de los procesos de negocio se vuelve más compleja, dado que es necesario determinar cuál de las fuentes es la que contiene la información más actualizada. Además, la integración de estos ‘silos’ no es sencilla, y un cambio en cualquiera de ellos puede tener implicaciones en la arquitectura empresarial en su conjunto.

Si no se cuenta con una versión consistente de los datos maestros, es difícil tener un entendimiento profundo de los mismos.

Lo ideal sería que existiera una fuente única de la verdad, es decir, un lugar único donde se almacenen y administren los datos maestros, asegurando que éstos sean correctos, completos y consistentes. Todas las operaciones sobre los datos deberían realizarse sobre esta fuente única, que debería garantizar la seguridad y privacidad de los datos maestros que almacena. Esta fuente autoritativa de los datos maestros debería ser capaz de exponer la información y permitir su administración mediante servicios que faciliten integración con otras aplicaciones y canales, además ser flexible para que sea capaz de evolucionar junto con la organización. El modelo de datos debería poder extenderse de forma transparente e idealmente sin afectar a las aplicaciones que se integran a la fuente única. MDM tiene como objetivo convertir todas estas ideas en realidad.

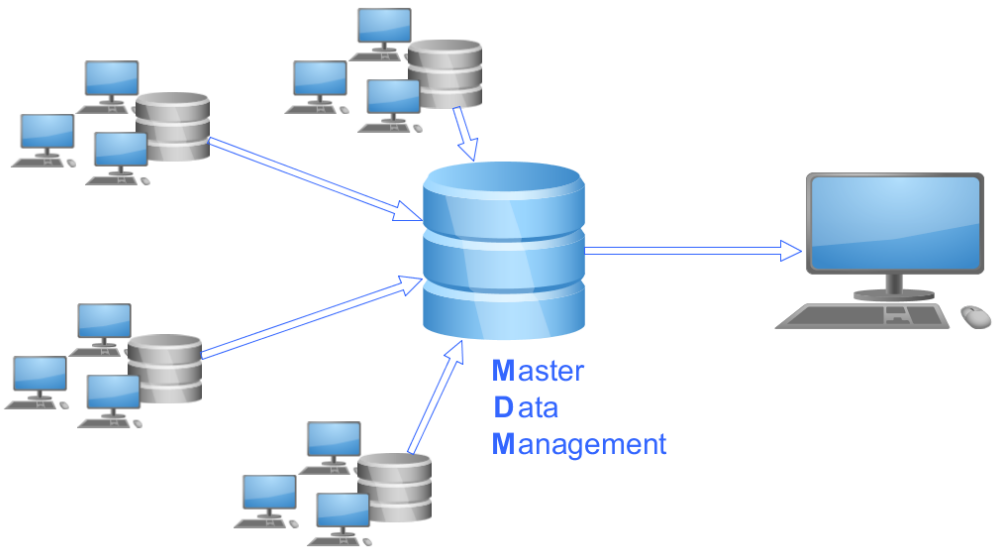


fig. 2 Manejo, limpieza y consolidación de datos con MDM

A modo de resumen, podemos decir que el MDM:

- Anula la información redundante.
- Evita diferentes procedimientos para el mantenimiento de datos maestros (como la ausencia de equipos de administración centralizada).
- Minimiza la frecuencia de intervenciones manuales (que terminan conduciendo a errores y necesitando de ajustes).
- Anula la ausencia de procedimientos que aporten visión unificada y/o global.

- Elude la información inconsistente en sistemas que disponen de información agregada o consolidada (que originan problemas de calidad de datos).
- Evita la duplicidad de procedimientos de obtención de información.
- Minimiza el riesgo operacional en el entorno analítico.

2.3.2 Repercusión

Una solución MDM no deja de ser una inversión. Tener "múltiples versiones de datos verdaderos" conduce con toda probabilidad a operaciones empresariales ineficientes, a la toma de decisiones incorrectas, previsiones poco precisas, y a la incapacidad para adherirse a las directrices regulatorias, y otros problemas que pueden obstaculizar el funcionamiento de la empresa, consumir recursos y reducir la ventaja competitiva. [9] Además, el solo hecho de mantener múltiples aplicaciones con su propia base de datos que almacene información maestra implica mayores costos en equipos y licencias.

La tecnología MDM ayuda a mejorar los niveles de servicio de TI al mismo tiempo que se reducen costos, porque asegura que los datos críticos para una empresa son correctos, completos y consistentes mientras se crean, modifican o consumen por procesos de negocio internos y externos, aplicaciones y usuarios.

En el pasado ya se intentó lograr los beneficios de consolidar los datos maestros en un repositorio central, pero solamente mediante una solución de MDM integrada, flexible, multi-dominio, multi-función y orientada a servicios es posible obtener beneficios tangibles.

Se puede decir que MDM es el pináculo de las tecnologías de integración de datos, ya que tiene la capacidad de trabajar con múltiples dominios, provee mejores mecanismos para el mantenimiento de la calidad de los datos y proporciona su funcionalidad en forma de servicios, que permiten integrarla con mayor facilidad a ambientes heterogéneos.

2.3.3 Éxito de implantación

Muchas iniciativas de MDM están destinadas a fracasar, porque las empresas las plantean erróneamente como un proyecto individual a corto plazo. La gestión de datos maestros debe entenderse como una estrategia de largo alcance y a largo plazo que combine políticas y procedimientos con soluciones de tecnología avanzadas. También es imprescindible que los responsables de los departamentos de TI y las áreas funcionales clave del negocio estén involucrados de forma activa para que la estrategia de gestión de datos maestros tenga éxito.

2.3.4 Soluciones similares

A continuación se comparará la tecnología MDM con otro tipo de soluciones, enfatizando en las diferencias y el beneficio de utilizar MDM.

2.3.4.1 Operational Data Store

De acuerdo a Bill Inmon, un Operational Data Store (de ahora en adelante ODS) es "una estructura integrada, orientada a un tópico, volátil (incluye actualizaciones) que mantiene el valor actual de los datos". El ODS es alimentado por las aplicaciones operacionales a partir de programas de integración y transformación. El ODS, a su vez, alimenta a un Datawarehouse. Estas características son semejantes a las de un MDM implementado con el estilo de consolidación. La diferencia radica en los servicios que ofrece una solución de MDM: mientras que el ODS es una base de datos usada para agregar información con un objetivo en particular, el MDM proporciona servicios de acceso, gobernabilidad y administración de datos. El enfoque de proporcionar la funcionalidad de MDM en forma de servicios facilita la integración de la solución en un ambiente SOA. Además, mientras que un ODS está enfocado principalmente a alimentar soluciones analíticas, una solución MDM implementada con el estilo adecuado (registro, coexistencia o transactional hub) puede soportar el método de uso operacional.

2.3.4.2 Customer Information File

Un Customer Information File (de ahora en adelante CIF) es un sistema utilizado primordialmente en el ámbito bancario, en donde se almacena la información de los clientes y sus cuentas. Esto permite entender al cliente como una sola entidad, en vez de verlo en términos de cuentas separadas. Se actualiza frecuentemente, en ocasiones a diario, según ocurren movimientos en las cuentas de los clientes. Cuando se usa en Marketing, un CIF puede ayudar a identificar oportunidad de cross-sell y up-sale, producir listas de clientes para envío de publicidad por correo, así como analizar la rentabilidad de las relaciones con los clientes []. Además de satisfacer la funcionalidad mencionada, una solución de MDM puede manejar la calidad de los datos, manteniéndolos limpios y estandarizados, de manera que la información provista sea confiable. Ese no es el caso de un CIF que al no contar con un proceso proactivo para la de-duplicación de datos, puede proporcionar información "sucia" a los sistemas que alimenta.

Mientras que un CIF está enfocado en el dominio de clientes, una solución MDM puede manejar distintos dominios. Otra de las ventajas del uso de MDM con respecto a CIF es la capacidad de hacer operacional la información almacenada en la solución, siempre que se haya utilizado el estilo de implementación adecuado.

2.3.4.3 Data Warehousing

Un Data Warehouse surge de la necesidad de contar con un repositorio único, confiable, consolidado e integrado de datos para fines de reporteo y análisis. Dado este propósito, los Data Warehouses frecuentemente se diseñan utilizando un modelo basado en dimensiones. En esto radica la principal diferencia con una solución MDM. Si bien el método de uso analítico está enfocado a proporcionar información completa, correcta y consistente a sistemas analíticos, MDM puede además hacer operacional la información que contiene (de acuerdo al estilo de implementación que se haya utilizado). El hacer operacional la información implica que el modelo de datos de una solución MDM esté diseñado para mantener la integridad y no redundancia de los datos. A diferencia de los casos anteriores, en que una solución de MDM puede sustituir a un ODS o a un CIF, MDM y Data Warehousing no son "excluyentes" entre sí, más bien tienen distintos propósito e incluso se puede considerar que se complementan.

3. Caso de estudio

Una organización (simulada por un compañero de la empresa donde se ha desarrollado el presente trabajo), a quien nombraremos a partir de ahora como 'cliente', tiene (como ya hemos expuesto en apartados anteriores) sus datos repartidos en 5 sistemas de información diferentes que se comunican entre sí, cada uno de los cuales dispone de su propia base de datos.

El caso de estudio se centra en el ciclo de vida de un activo, desde que se solicita el pedido, pasando por su instalación y planificación de las tareas de mantenimiento, hasta su desmantelamiento.

El cliente nos pide un sistema de información capaz de aunar toda la información que maneja, para poder localizar de forma fiable el momento óptimo de comienzo de tareas de mantenimiento, así como el de su desmantelamiento y sustitución.

3.1 Sistemas involucrados

A continuación, se describen brevemente cada uno de los sistemas implicados.

3.1.1 Supply Chain Manager

El Supply Chain Manager (SCM de ahora en adelante) es un administrador de redes de suministro. Se encarga de la planificación, puesta en ejecución y control de las operaciones de la red de suministros de la organización.

Software disponibles: Navision, iTracker Hosted 3PL, Fishbowl Inventory, 3PL Warehouse Manager, Freightview, etc.

3.1.2 Enterprise Resource Planning

El Enterprise Resource Planning (ERP de ahora en adelante) es un sistema de planificación de recursos empresariales, encargado de la administración de recursos de la organización.

Software disponibles: Oracle JD Edwards, Microsoft Dynamics GP, SAP Business ByDesign, Sage ERP, etc.

3.1.3 Computerized Maintenance System

El Computerized Maintenance System (CMMS de ahora en adelante) se encarga de la gestión del mantenimiento asistido por ordenador. Contiene información de la organización y sus operaciones de mantenimiento para ayuda a que las tareas de mantenimiento se realicen de forma segura y eficaz.

Software disponible: emaint, Maintenance Connection, MaintenanceDirect, Hippo CMMS, etc.

3.1.4 Geographic Information System

El Geographic Information System (GIS de ahora en adelante) es el sistema de información geográfica. Su cometido es el de relacionar diferentes componentes que permiten la organización, almacenamiento, manipulación, análisis y modelización de datos vinculados a una referencia geográfica.

Software disponible: Maptitude, Map Business Online, PubWorks, Google, Bink, teleatlas, etc.

3.1.5 Content Management System

El Content Management System (CMS de ahora en adelante) es un sistema de gestión de contenido. Se utiliza para crear estructuras de soporte para la creación y administración de contenidos como, por ejemplo, las wikis.

Software disponible: Contentful, Wild Apricot, eDirectory, etc.

3.2 Flujo de trabajo inicial

El encargado de recursos inicia una solicitud de pedido en el ERP, definiendo para ello el/los elementos a pedir, el coste económico de los artículos del pedido (precio de adquisición), los datos del proveedor, del plazo esperado de entrega en adquisición (que puede afectar a la caducidad del pedido), el nombre del activo, persona que autoriza la compra, sistema de activos en el que se integra, fecha del pedido, etc. El sistema le asigna al pedido un número de pedido. Los datos del pedido se transmiten al SCM.

Cuando el pedido llega, este se recepciona por el encargado de recepciónes, generando un albarán interno en el SCM que lleva asociados todos los datos que se tenían ya del pedido, más una referencia interna del activo que ya está en posesión de la organización, la fecha de recepción y los de la persona que lo recibe.

El mismo usuario que recepciona el pedido en el SCM mete en el CMS la referencia interna del activo en cuestión para añadir a éste la documentación que le pueda afectar

(manuales, homologaciones del activo, regulaciones que lo afectan, certificaciones necesarias para su mantenimiento, vida útil esperada, vida máxima permitida por la ley, etc.).

La referencia interna del activo, su fecha de recepción, la persona que lo recibe y lu vida útil esperada se envían al ERP.

El ERP le asigna una cuenta contable, un centro de coste y un ritmo de amortización fiscal. Cuando llega la factura del proveedor, se graba la fecha de factura.

Inmediatamente después de la recepción o cuando llega el momento en que el activo se va a poner en funcionamiento, alguien lo coloca/monta/instala. En este punto, el SCM genera una serie de datos relativos al montaje del activo tales como el proveedor que lo instala, el coste de contratación del montaje, la persona que autoriza dicha contratación (en caso de que dicho montaje se lleve a cabo por personal externo), la fecha de inicio, la fecha prevista de finalización y la fecha de entrada en funcionamiento.

Una vez puesto en marcha el activo, se inserta en el GIS la referencia interna y el nombre del activo junto con las coordenadas de su ubicación física.

Se introduce en el CMS la documentación relativa a su montaje que le pueda afectar (contrato del montador, garantías de su montaje, etc.) asociada a la referencia interna del activo en cuestión.

Una vez que el activo ha entrado en funcionamiento, se le pasa al ERP la información relativa a los costes de contratación de montaje y del montador (esos costes incrementan su coste de adquisición tanto en su cuenta contable como en el valor a amortizar) y la fecha de entrada en funcionamiento para dar por comenzado su periodo de amortización.

En ese mismo momento, se le manda al CMMS la referencia interna del activo, su nombre y la documentación que le afecte (manual de uso, garantía, certificaciones necesarias para su mantenimiento, regulaciones que le afectan, vida real útil esperada, vida máxima permitida por la ley, etc.). Este sistema establece un calendario de trabajos a realizar de cara a su mantenimiento preventivo y un seguimiento del rendimiento objetivo del activo. También establece la criticidad del activo.

A medida que se van ejecutando las acciones de mantenimiento (tanto las programadas de mantenimiento preventivo como las imprevistas ocasionadas por averías o bajadas de rendimiento), el CMMS va guardando datos relativos a las fechas en las que se producen estas acciones, los que llevan a cabo el mantenimiento (subcontratas o personal propio),

el coste de la reparación y el impacto en la vida útil del activo de la avería (cambio en su vida útil real esperada).

A medida que las tareas de mantenimiento se van llevando a cabo, se va informando de los costes de reparación de los activos en cuestión al ERP.

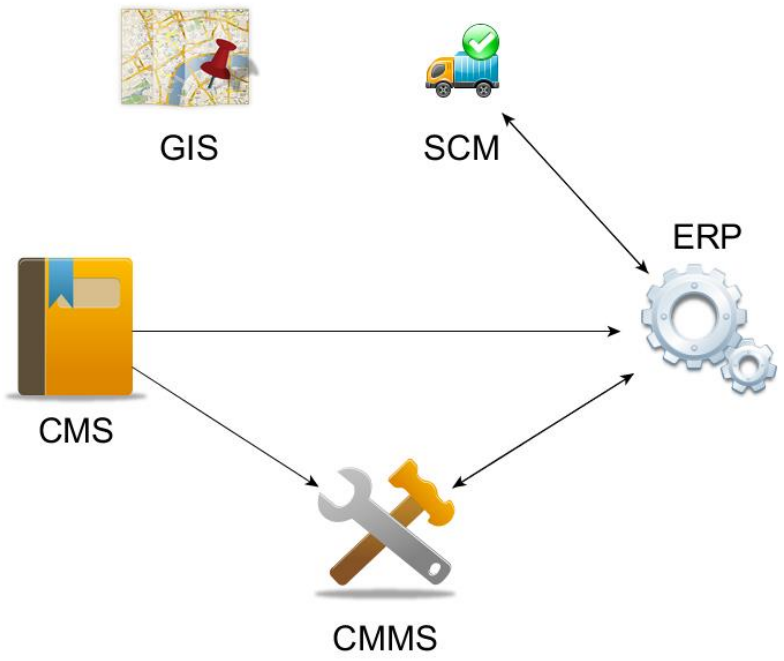


fig. 3 Diagrama de interrelación inicial entre sistemas

3.3 Funcionamiento tras la implantación

Toda la información generada hasta el momento de la implantación de la nueva solución se vuelca al sistema MDM en una carga inicial.

Desde el cuadro de mando (BI), con toda la información referente a los activos que recibe del MDM, como el rendimiento en producción y a la vida útil esperada tras las últimas operaciones de mantenimiento, se irán tomando decisiones acerca de la periodicidad de las tareas de mantenimiento o si se debe hacer alguna tarea extra imprevista, hasta que se llegue a la conclusión de que hay activos totalmente amortizados, en cuyo caso se puede mandar al SCM la orden de retirada/sustitución.

En ese último caso se agendan las tareas de desmantelamiento del activo y se generan datos tales como los proveedores de servicios de desmantelamiento, costes de desmantelamiento, costes de gestión de los residuos y valores residuales (si los hubiera).

Se pasan al ERP los datos del desmantelamiento del punto anterior, que imputa los costes al centro de coste correspondiente al activo en cuestión.

Toda la información que se vaya generando se mandará de forma periódica al MDM.

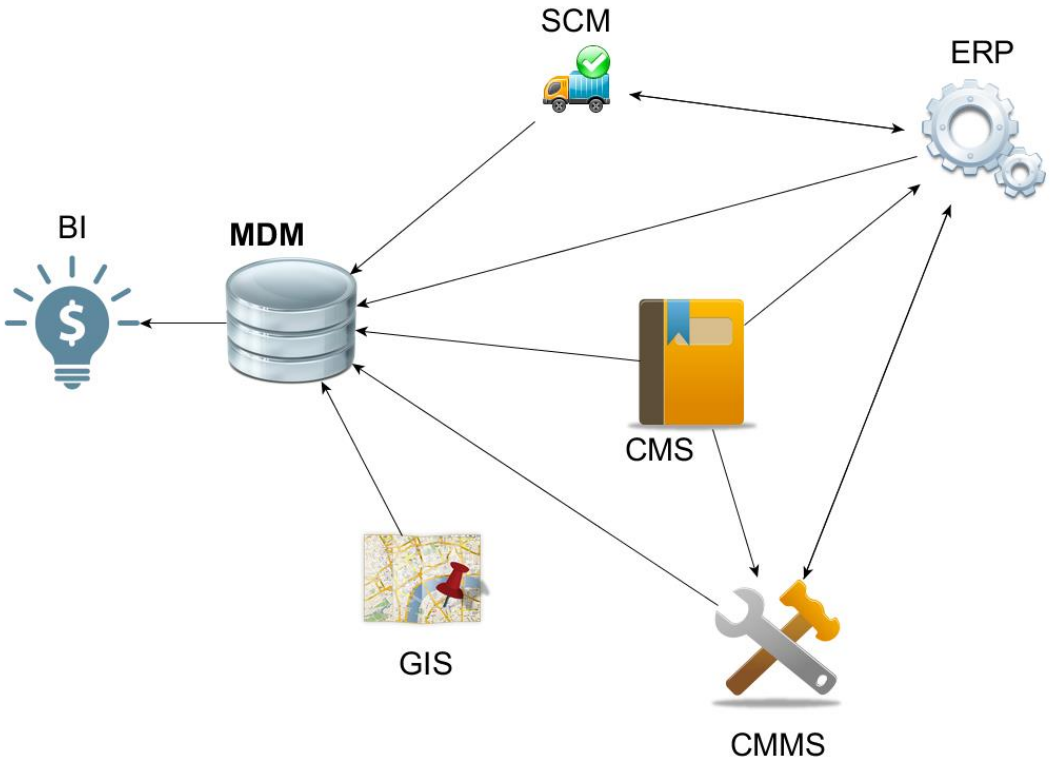


fig. 4 Diagrama tras la implantación del sistema MDM junto con el sistema BI

4. Metodología

En este apartado se exponen todas las herramientas y métodos utilizados en este trabajo para su realización y puesta en funcionamiento. Muchas herramientas sólo se han utilizado en fases tempranas del trabajo, con el propósito de interiorizar un enfoque global a las posibles soluciones a aplicar ya que, en proyectos reales, cada cliente tendrá unas características y un entorno de trabajo distintos a los que habrá que adaptarse.

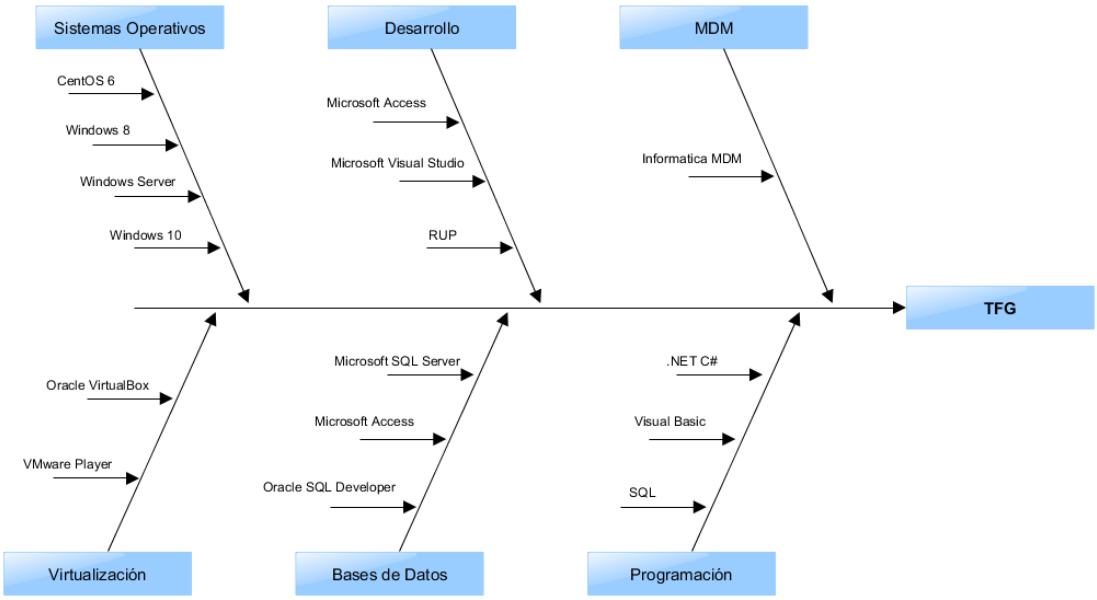


fig. 5 Diagrama de Ishikawa de la metodología empleada

4.1 Arquitectura Cliente-Servidor

La implementación de la solución propuesta se basa en una estructura cliente-servidor, en el que los 5 sistemas de la organización vuelcan datos a un servidor, mientras que el cuadro de mando la extrae para su consulta.

4.1.1 Definición

En la arquitectura cliente-servidor, un servidor es un proveedor de recursos que se encarga de dar respuesta a una petición realizada por unos demandantes (clientes).

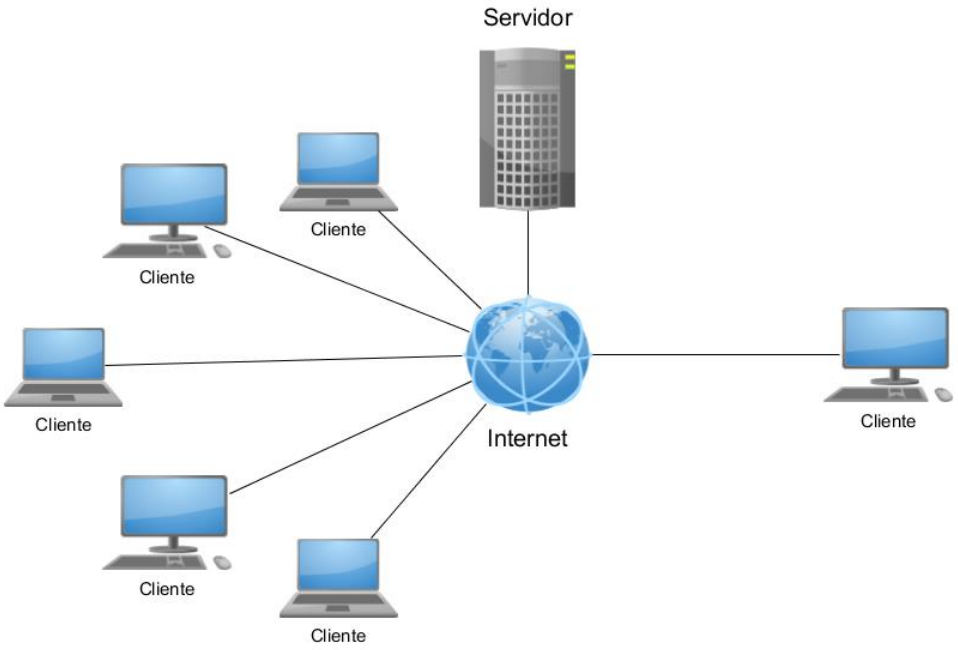


fig. 6 Diagrama de la aquitectura Cliente-Servidor

4.1.2 Componentes y funcionamiento

Todo sistema basado en una arquitectura cliente-servidor tiene una serie de componentes en común:

4.1.2.1 Servidor

Un servidor (o back-end) es el encargado de proporcionar una serie de servicios a quien los solicite. Un servidor se encarga de atender a múltiples clientes, aceptar y procesar sus requerimientos, y procesar la lógica y las validaciones correspondientes.

En este caso, el servidor es una máquina con sistema operativo Windows Server donde se aloja un sistema de gestión de bases de datos, SQL Server, junto con el centro de actividad (o hub) de MDM.

4.1.2.2 Cliente

El cliente (o front-end) reclama unos servicios en base a unos requerimientos formulados por el usuario. Normalmente, están constituidos por una interfaz de usuario y sus funciones suelen estar relacionadas con el despliegue y manipulación de datos. Además, un cliente se encarga de:

- Interactuar con el usuario
- Procesar la lógica de la aplicación
- Hacer requerimientos a la base de datos

- Dar formato a los resultados

En este caso, los clientes son los sistemas CMS, SCM, CMMS, ERP, GIS (aplicaciones de usuario que solicitaran la inserción de datos periódicamente) y BI (basado en lógica de negocio, que solicitará acceder y consultar a dichos datos para efectuar los cálculos necesarios para su interpretación).

4.1.2.3 Middleware

El middleware es un software de interconexión que facilita la interacción entre el cliente y el servidor. Incluye elementos como las pilas de comunicación, los directorios distribuidos, los servicios de autenticación, las llamadas a procedimientos remotos y el manejo de colas.

En este trabajo, no se ha tratado este tema.

4.1.3 Características

La arquitectura aporta las siguientes características:

- Alto grado de disponibilidad y fiabilidad.
- Recursos centralizados.
- Administración centrada en el servidor.
- Escalabilidad.

4.1.4 Justificación

La razón de la utilización de esta arquitectura es, de hecho, uno de los puntos fuertes de la tecnología MDM: exponer la información y permitir su administración mediante servicios, lo que facilita la integración con otros canales o aplicaciones y ofrecer flexibilidad para que el sistema pueda evolucionar junto con la organización.

4.2 Softwares de Virtualización

Un software de virtualización se utiliza para simular una máquina física con todas sus características y requerimientos (CPU, BIOS, memoria RAM, tarjeta de red, disco duro, etc.).

fig. 7 Concepto de virtualización de máquinas

La justificación de la utilización de máquinas virtuales radica en la necesidad de simular una máquina servidor para proveer los servicios que van a ser proporcionados por la herramienta de MDM y demandados por los sistemas de información clientes. Esta solución nos permite tener un servidor sobre el que tendremos control total, a la vez que capacidad de clonación o copia (permitiendo la creación de un entorno de test) y la reducción de hardware y sus costes asociados.

Cada software de virtualización se ha empleado en momentos distintos, para así poder hacer una comparativa y poder determinar qué software se acoplaba mejor a las características del trabajo.

4.2.1 VMware Player

VMware Player es la versión gratuita (para uso no comercial) de VMware y se utilizó en las primeras fases del trabajo para crear una máquina virtual con sistema operativo CentOS 6.

Su principal desventaja es que, al ser una versión gratuita, tiene una funcionalidad limitada.

4.2.2 Oracle VirtualBox

El sistema de virtualización de Oracle, VirtualBox, es un software bajo licencia open-source. Las principales diferencias por las que se optó por utilizar esta solución son las siguientes:

- Facilidad a la hora de crear clones de las máquinas virtuales
- Soporta múltiples tipos de imagen de disco.
- Capacidad de hacer múltiples Snapshots
- Permite crear máquinas en un host y arrancarlas en otros distintos.

Se creó una máquina virtual con sistema operativo Windows Server 2008 R2, que es la máquina que finalmente se utilizó para crear el servidor, con los siguientes componentes:

- 2 CPU
- Memoria de 4 GB
- Chipset basado en PIIX3 (Intel 82371)
- Adaptador de red conectado a NAT

4.2.3 Otras alternativas

Otras soluciones de virtualización podrían ser Microsoft Virtual PC (con una lista limitada de hosts soportadas) o VMLite Workstation (basado en VirtualBox).

4.3 Interconexión de redes

Al utilizar una arquitectura cliente-servidor, hay que tener muy en cuenta la importancia de una correcta elección de las herramientas y de la configuración de la conectividad con la red, ajustada a los requerimientos del problema.

4.3.1 Adaptador Puente

Para conectar el adaptador de red de la máquina virtual se optó inicialmente por usar un adaptador puente (o bridge).

Esta solución crea una tarjeta de red virtual en el anfitrión que intercepta el tráfico de red y puede inyectar paquetes, de manera que el huésped se configura como si estuviera conectado por un cable a la tarjeta de red virtual del anfitrión.

4.3.2 NAT

Utilizar NAT (Network Address Traslation o traducción de direcciones de red) ha sido la solución final para conectar el adaptador de red de la máquina virtual.

Se basa en la traducción de direcciones IP de acuerdo con una tabla de equivalencias. La traducción suele hacerla el encaminador que conecta la red al exterior. Se puede utilizar para:

- 'Extender' el rango de direcciones disponibles en una red.
- Conectar a Internet redes IP que utilizan rangos privados.

Hay dos tipos dependiendo de los campos que se modifican: NAT básico y NAPT (Network Address Port Translation). Este último es el que se ha utilizado para que la máquina virtual sea visible desde el resto de máquinas de la red y sus servicios puedan

ser accesibles (en este caso, el que actuará como encaminador y, por tanto, hará las labores de traducción será el motor de red de VirtualBox). Consiste en modificar tanto la dirección IP como el número de puerto.

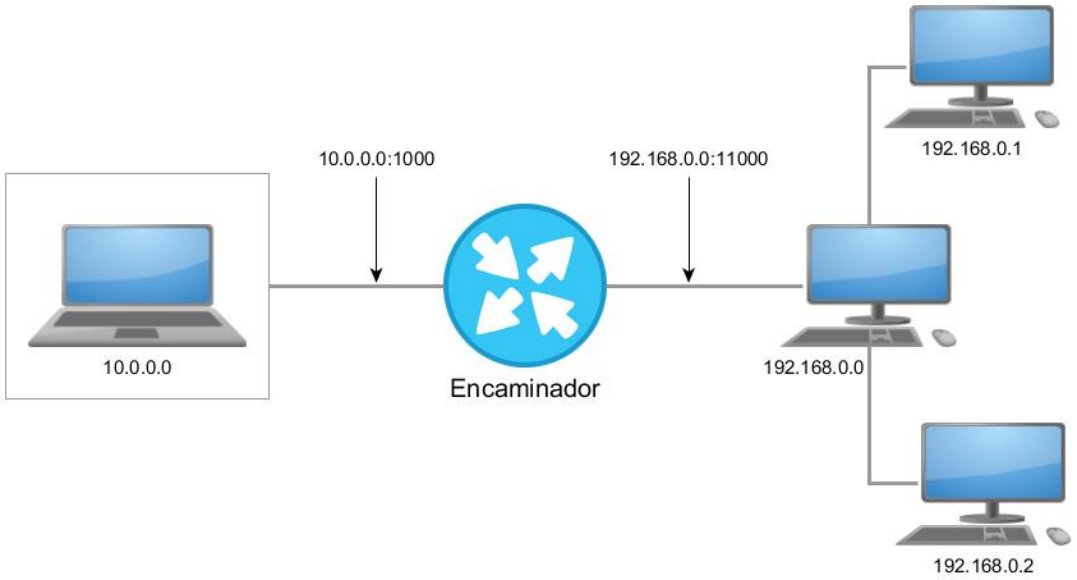


fig. 8 Ejemplo de uso de NAT

4.4 Sistemas operativos

El cambio o la utilización simultánea de distintos sistemas operativos responden a varias circunstancias como requerimientos de la empresa, requerimientos del problema o adaptación a nuevos escenarios derivados de cambios ajenos a este proyecto.

4.4.1 CentOS 6

CentOS (Community ENTerprise Operating System) es un sistema operativo de tipo empresarial y de código abierto basado en la distribución Red Hat Enterprise Linux.

Fue la primera alternativa de sistema operativo que se utilizó para el servidor.

4.4.2 Windows Server 2008 R2 Enterprise

Windows Server es el sistema operativo de Microsoft diseñado para servidores. Esta versión permite crear sesiones de usuarios en paralelo y tiene mejoras en el rendimiento de la virtualización. Otra característica a destacar es el cierre limpio de servicios.

Es el sistema operativo que se ha utilizado finalmente para nuestro servidor.

4.4.3 Windows 8.1

Es el primer sistema operativo de la máquina anfitriona o host. Se optó por un sistema Windows porque en todo momento se ha orientado el trabajo al usuario final.

Los principales cambios con respecto a otras versiones han sido principalmente en la interfaz de usuario, la cual eliminaba el menú de inicio presente desde Windows 95 y se adecuaba más para su uso con pantallas táctiles, aunque se seguía pudiendo usar el ratón y el teclado. Esto se puede considerar un aspecto negativo si, como en este caso, se utiliza como estación de trabajo.

4.4.4 Windows 10

Segundo sistema operativo de la máquina anfitriona tras la actualización que publicó Microsoft en Julio de 2015 (cabe señalar que la máquina anfitriona ha sido también el equipo en el que se ha llevado a cabo el grueso de trabajo durante todo el desarrollo).

Esta actualización recupera el menú de inicio, mantiene los mismos requisitos que su predecesor y se adapta mejor a cada espacio de trabajo. Una de las características más interesantes, desde el punto de vista de la eficiencia en el trabajo, es la posibilidad de crear múltiples escritorios.

El único problema que repercutió en el trascurso del trabajo fue la imposibilidad de utilizar un adaptador puente en VirtualBox.

4.5 Metodologías Ágiles

El uso de Metodologías Ágiles en el desarrollo, concretamente la metodología Extreme Programming o XP, está indicado para grupos de trabajo pequeños y viene dado por la necesidad de colaborar con el cliente activamente y no ceñirse a un contrato preestablecido, ya que habrá una probabilidad alta de que haya que responder a cambios.

Pese a que habrá un modelado y una elaboración de documentación del sistema, ya que se parte de unos requisitos iniciales del cliente, prevalecerá desarrollar un software que funcione e ir adecuándose a las nuevas necesidades que vaya trasmitiéndonos el cliente.

Los principios que nos han llevado a utilizar estas metodologías son:

- Tener satisfecho al cliente mediante tempranas y continuas entregas de software utilizable.
- Construir el proyecto en base a motivaciones personales.
- Disminuir el tiempo entre entregas de software desarrollado.

- No poner problemas a la hora de introducir cambios, lo que permitirá al cliente ser competitivo.
- Involucrar al cliente en el proyecto como parte del equipo.
- Promover un desarrollo llevadero.

4.6 Lenguajes de programación

Los diferentes escenarios que se plantean en el presente trabajo han requerido que se utilicen distintos lenguajes de programación, dependiendo del ámbito del problema.

4.6.1 Microsoft .NET C#

.NET es un "framework" o plataforma de desarrollo de software de Microsoft que está planteada para facilitar rapidez en el desarrollo de aplicaciones, y que hace énfasis en la transparencia de redes, ya que provee de un extenso conjunto de soluciones predefinidas.

C# es el lenguaje que nació para ser utilizado en .NET. Orientado a objetos (admite los conceptos de encapsulación, herencia y polimorfismo) y con seguridad de tipos, su sintaxis básica deriva de C/C++ y es muy similar a la de Java. La sintaxis de C# simplifica muchas de las complejidades de C++ y proporciona características eficaces tales como tipos de valor que admiten valores NULL, enumeraciones, delegados, expresiones lambda y acceso directo a memoria, que no se encuentran en Java. Se puede usar para crear aplicaciones cliente de Windows, servicios Web XML, componentes distribuidos, aplicaciones cliente-servidor, aplicaciones de base de datos, etc.

Con este lenguaje y la plataforma de desarrollo Visual Studio (como se explica en el punto 4.8.1), se ha implementado el cuadro de mando del sistema BI.

4.6.2 Visual Basic

Este lenguaje de programación, dirigido por eventos, ha sido empleado en la simulación de las fuentes que proveen datos al MDM. Concretamente, es con lo que se han programado los eventos disparados por la pulsación del botón de carga de datos desde los formularios implementados en Access, ya que es la forma que tiene Access de programar los eventos.

4.6.3 SQL

SQL (Structured Query Language) es un lenguaje de tipo declarativo (especifica qué quiere, pero no cómo conseguirlo) de alto nivel, empleado para hacer consultas sobre bases de datos relacionales (crear, leer, consultar o eliminar). Este lenguaje ha sido el más empleado, ya que todas las bases de datos empleadas son de tipo relacional.

4.7 Sistemas de gestión de bases de datos relacionales

Los sistemas de gestión de bases de datos (SGBD de hora en adelante) permiten el almacenamiento, codificación y extracción de información de las bases de datos. También proporcionan métodos para mantener la integridad, administrar el acceso de usuarios y recuperar la información en caso de fallo. Suelen incluir módulos gráficos que permitan presentar la información que contienen.

En nuestro caso, los SGBD son relacionales y se accederá a su información a través de lenguajes de interrogación y, más concretamente, SQL.

4.7.1 Oracle Database

Se utilizó en fases tempranas del trabajo. Desarrollado por Oracle Corporation, se considera uno de los SGBD más completos (su egemonía en el mercado era casi total hasta la llegada de Microsoft SQL Server o MySQL), ya que da soporte de transacciones, soporte de escalabilidad y soporte multiplataforma.

4.7.2 Microsoft Access

Está incluido en el paquete ofimático Microsoft Office y es la solución propia de bases de datos relacionales de Microsoft.

Se ha utilizado para crear las bases de datos propias de cada uno de los 5 sistemas (ERP, SCM, CMMS, GIS y CMS) que se han simulado, ya que únicamente estaba el propósito de simular dichos sistemas de forma rápida. Cada tabla se puede crear de forma sencilla a través de un asistente.

En el apartado 4.8.2 se explicarán otro tipo de utilidades de esta herramienta.

4.7.3 Microsoft SQL Server

Desarrollado por Microsoft, algunas de sus características más reseñables son que da soporte de transacciones, soporta procedimientos almacenados, permite trabajar en modo cliente-servidor y permite administrar información de otros servidores.

Este es el SGBD que se ha utilizado finalmente en el servidor y que será en donde se almacenen todas las tablas y datos que se definan y se incluyan en el sistema MDM.

4.8 Software de desarrollo

Los softwares de desarrollo contribuyen a una visión general del proyecto y facilitan la tarea del desarrollador. Permiten editar código fuente, compilarlo (y/o interpretarlo), agilizan el trabajo mediante herramientas de automatización y ayudan en la depuración.

4.8.1 Visual Studio

Visual Studio es el entorno de desarrollo para Windows que soporta múltiples lenguajes, como Java, Python y C#, entre otros. Permite crear cualquier aplicación en cualquier entorno que soporte la plataforma .NET.

Proporciona un editor de código avanzado, cómodos diseñadores de interfaz de usuario, depurador integrado y numerosas herramientas más para facilitar el desarrollo de aplicaciones basadas en lenguajes como C# y .NET Framework.

Ha sido utilizado en el desarrollo del cuadro de mando, ya que esta aplicación ha sido planteada para ser utilizada en entornos Windows.

4.8.2 Microsoft Access

Access también ofrece la posibilidad de desarrollar formularios simples de forma rápida y sencilla. Se ha utilizado para simular las acciones de los sistemas (ERP, SCM, CMMS, GIS y CMS), que proveen los datos al MDM. No solo se han desarrollado los formularios para crear, leer, actualizar y eliminar registros, sino que también se ha añadido, para cada uno, la funcionalidad de comunicarse con el servidor para volcar datos en el sistema MDM.

No entra dentro de los objetivos de este TFG profundizar en los distintos sistemas que se han nombrado, simplemente existía la necesidad de simularlos de forma que no nos ocupara demasiado tiempo, por lo que este software se consideró una opción ideal.

4.9 MDM

En este apartado se detallará la herramienta utilizada para la implementación del sistema MDM, además de los pasos a seguir en su implementación para un funcionamiento correcto.

En la siguiente figura se muestra, de forma conceptual, el proceso de implementación del sistema MDM en base a los requerimientos iniciales.

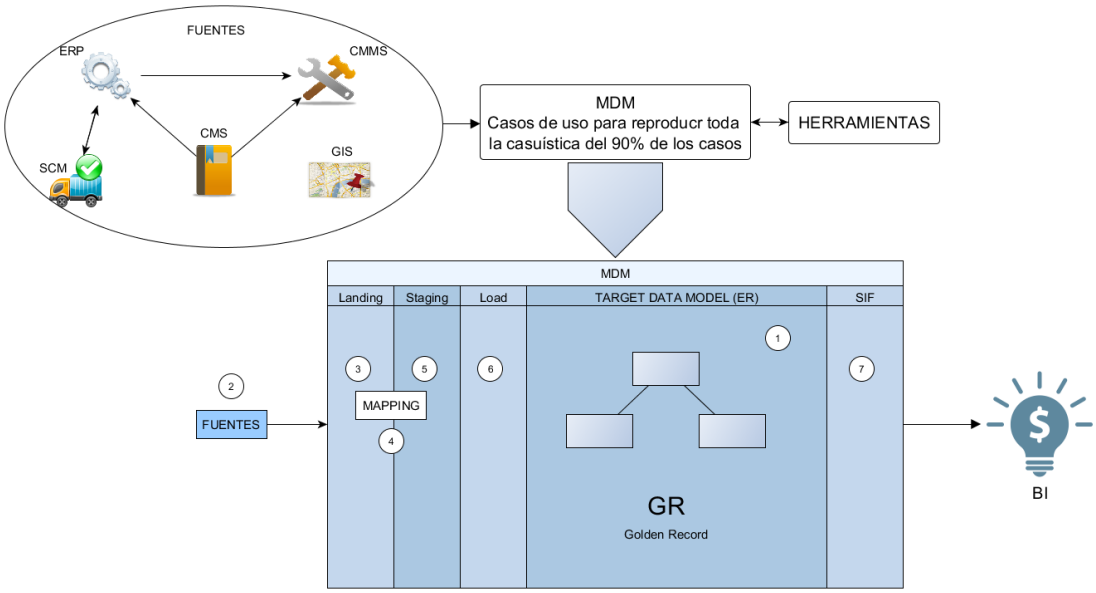


fig. 9 Proceso de imlementación

4.9.1 Informatica MDM

Esta herramienta es la que utilizaremos para implementar el sistema MDM.

"Informatica MDM es el primer sistema de MDM multidominio que permite manejar diversas entidades de datos de forma flexible, y que muestra de forma inteligente las relaciones entre dominios, todo dentro del mismo producto de MDM. Continúa manteniéndose a la vanguardia de las capacidades de los productos gracias a la adopción de MDM para Big Data, redes sociales y uso móvil. Cientos de clientes en múltiples sectores y diversos países han obtenido grandes ventajas de negocio a través de la reducción del riesgo y del plazo de amortización", afirma Ravi Shankar, vicepresidente de marketing de productos de MDM en Informatica.

Informatica MDM resuelve con rapidez los problemas de negocio inmediatos con una capacidad de gestión de datos maestros flexible y permite mejorar la solución inicial para resolver problemas de negocio adicionales mediante el uso de las inversiones y competencias tecnológicas existentes, al mismo tiempo que devalúa el riesgo del proyecto con una solución de MDM demostrada.

Informatica MDM permite:

- *Inicio rápido*: Una puesta en marcha rápida para numerosas implementaciones complejas y de gran tamaño mediante soluciones de serie. Un inicio rápido con una capacidad de MDM orientada al cliente, al producto o a cualquier otro dominio.

- *Crecimiento rápido*: Una vez que se han definido el modelo y las reglas, se puede añadir cualquier fuente, entorno alojado, dominio de datos, relaciones de datos, etc. sin apenas cambios y solo con un esfuerzo incremental. Una única gestión permite un uso en cualquier lugar.
- *Devaluación del riesgo*: Informatica MDM es una solución utilizada por cientos de clientes en numerosos sectores y países. Estos clientes utilizan Informatica MDM para resolver problemas complejos mediante el uso de varios dominios de datos. Informatica MDM permite gestionar cientos de millones de registros.
- *Trazabilidad*: La estructura de Informatica MDM permite la trazabilidad hacia atrás, lo que permitirá la resolución de errores, a la vez que garantizará la calidad de la información.

4.9.2 Plataforma

El trabajo de definición del sistema de MDM se llevará a cabo en lo que se denomina Hub Console, que es la herramienta gráfica de Informatica.

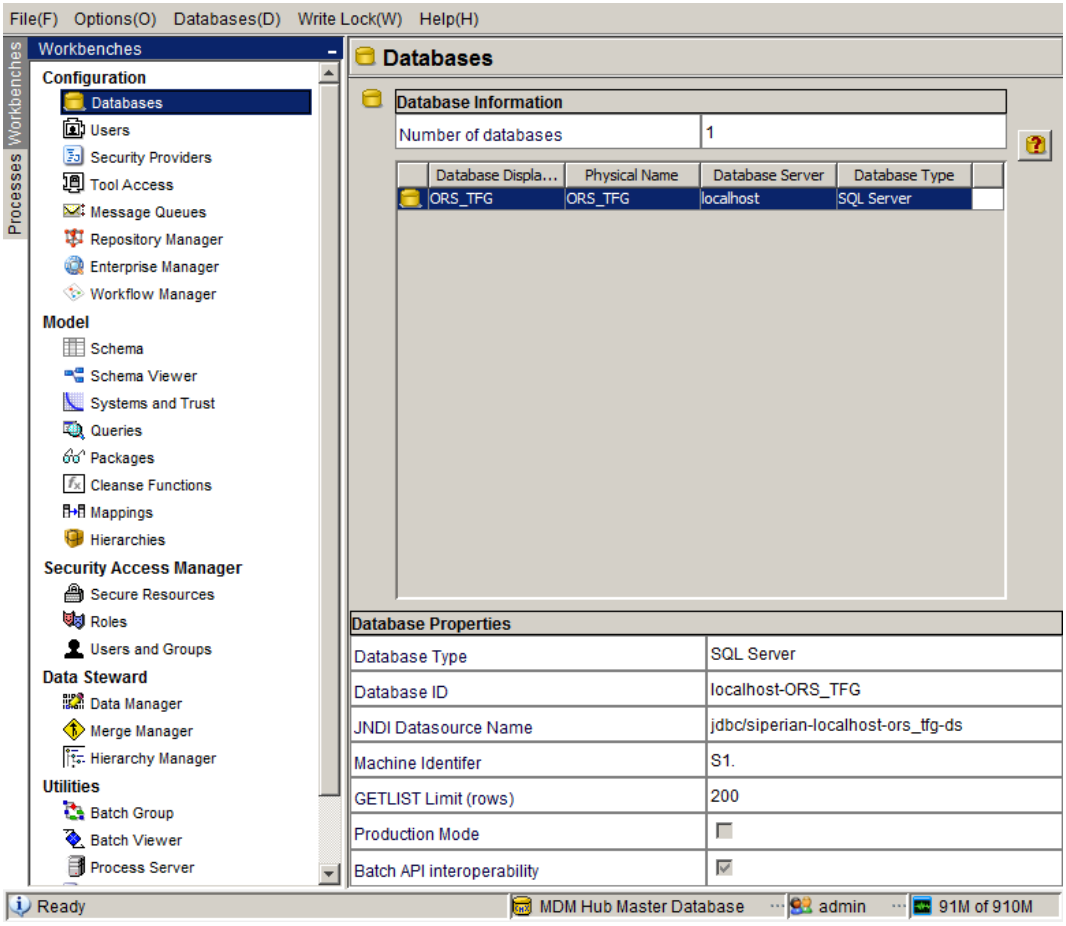


fig. 10 Hub Console Informatica MDM

4.9.3 Target Data Model

Es el modelo de datos que servirá de destino a todos los datos que procedan de las fuentes. El primer paso a realizar en la implantación de este sistema será crear un diagrama entidad-relación en base a los requerimientos iniciales del cliente. Cualquier duda ante el modelo (objetos, relaciones, listas de valores, etc.) se deberán consultar con el cliente.

Una vez hecho esto, se deberán crear las tablas en la plataforma del MDM dentro del apartado de 'Base Objects', junto con sus relaciones. La información que llegue a almacenarse en estas tablas serán los 'Golden Records', que es el nombre que se le da a los registros que se han considerado como la mejor versión de los datos.

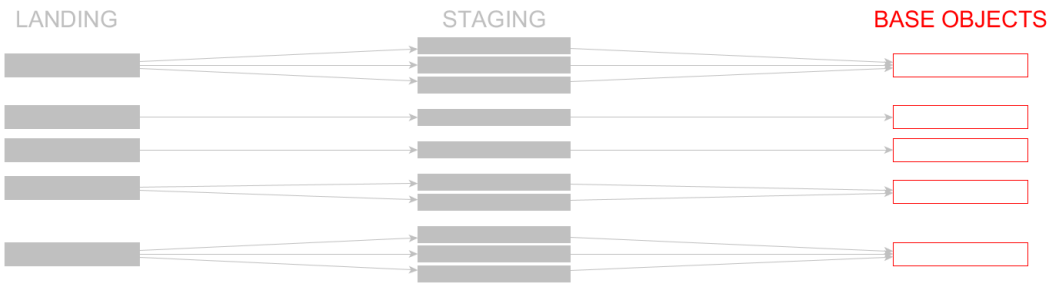


fig. 11

4.9.4 Tablas Landing

El segundo paso será crear las tablas 'Landing'. Estas tablas son en donde 'aterrizarán' los datos provistos por las fuentes.

Son un reflejo directo de las tablas de los Base Objects, es decir, habrá el mismo número de tablas y, en cada tabla, los campos correspondientes. Al ser tablas previas al tratamiento de datos, no habrá relación alguna entre ellas ni ningún tipo de restricción, pero sí deberán definirse los tipos.

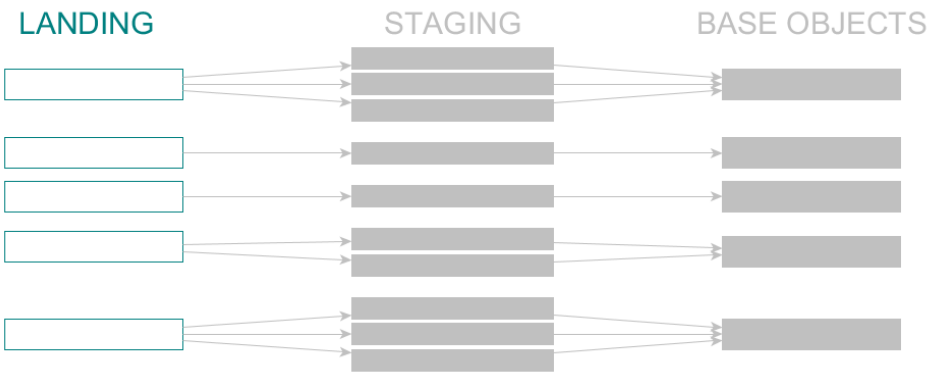


fig. 12

4.9.5 Mappings

Al definir los Mappings, se está estableciendo la relación que hay entre una tabla Landing y una Staging, es decir, para una fuente determinada, se deberá indicar explícitamente qué datos de una Landing provee y con cuales se corresponden en la tabla Staging.

A su vez, deberemos crear funciones que nos permitan identificar otras versiones del mismo dato ya insertadas por otros sistemas, ya que según el método de actualización escogido, servirá para modificar y actualizar registros insertados con anterioridad. Además, en los Mappins se disponen otras opciones de 'transformación', ya que cada unión entre columnas que se hace en los Mappins se pueden añadir funciones, constantes, componentes de ejecución condicional y combinaciones de todos los anteriores.

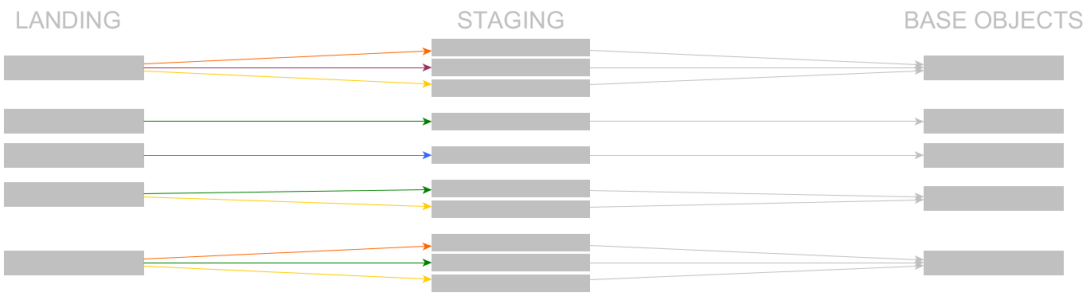


fig. 13

4.9.6 Tablas Staging

Su definición es el siguiente paso que se realiza. Una tabla Staging es un subconjunto de una de las tablas de los Base Objects, el cual deberá ser provisto de una de las fuentes.

Una Staging puede referirse solo a un Base Object, pero un Base Object puede estar referido por varias Stagging, y lo mismo pasa con las Landing.

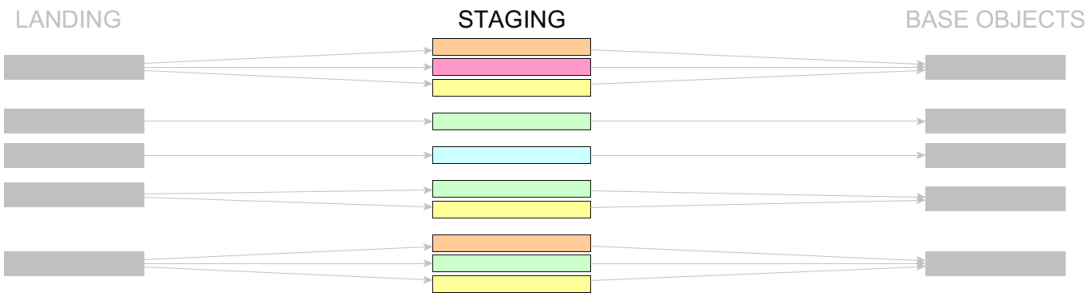


fig. 14

4.9.7 Batch Groups

Un Batch Group es un grupo tareas de carga de datos (desde las Landing a los Base Objects, pasando por los Mapping y las Stagins). En cada Batch Group se agrupan las tablas que no dependen entre sí, es decir, que ninguna tabla provea un dato a otra en el mismo Batch Group. También es importante que cada Bach Group esté en el orden correcto, para que, al ejecutarlos secuencialmente, no se creen errores de dependencia.

4.9.8 Actualización por Last Update Date

El modo de actualización por defecto y el que hemos elegido para el presente trabajo es por Last Update Date (LUD de ahora en adelante). Este método actualiza un campo de un registro, ya sea provisto por la misma fuente que lo insertó o por otra distinta, por la última fecha de modificación de su sistema de origen.

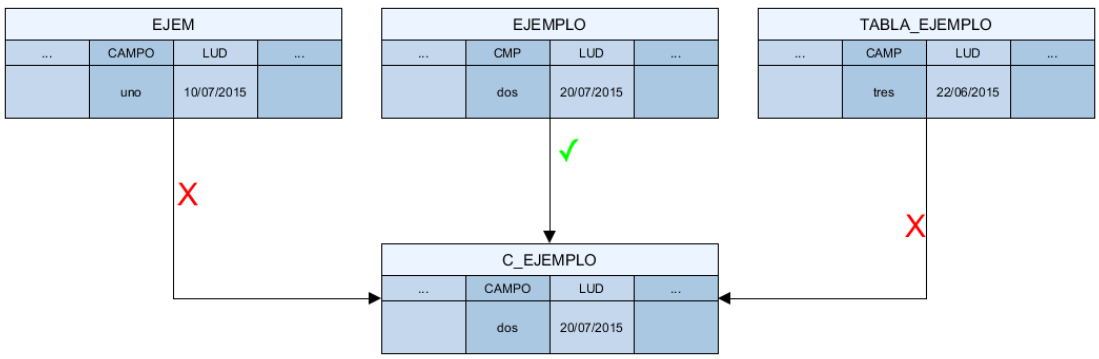


fig. 15 Concepto de actualización por LUD

4.10 Ilustración

Las tareas de diseño han sido un punto importante tanto en el planteamiento y diseño inicial de la solución al problema planteado como en la redacción de la memoria.

4.10.1 yEd

Tanto para las tareas de ilustración de conceptos y esquemas contenidos en la memoria para facilitar la comprensión de los conceptos expuestos, como para la realización de diagramas utilizados en el diseño de la solución planteada, se ha optado por utilizar el software de diseño gráfico yEd.

Se trata de un software open-source o freeware, enfocado principalmente en la creación de diagramas, que tiene una gran variedad de gráficos y la posibilidad de insertar otros nuevos elegidos por el usuario.

4.10.2 UML

UML es un lenguaje estándar de modelado de sistemas software respaldado por el OMG (Object Management Group) y utilizado para realizar diagramas de casos de uso, diagramas entidad-relación, etc.

Se ha aplicado en diversas fases del proyecto, especialmente en la planificación previa al desarrollo del cuadro de mando.

4.11 Otras herramientas utilizadas

A continuación, se describen otras utilidades que se han empleado en algún momento del desarrollo del trabajo.

4.11.1 WinMerge

WinMerge es una herramienta de diferenciación y combinación de código abierto para Windows. WinMerge tiene la capacidad de comparar tanto carpetas como archivos, presentando las diferencias en formato texto.

Se ha utilizado, básicamente, para comparar versiones de código antiguas.

4.11.2 Notepad++

Es un editor de texto gratuito que soporta múltiples lenguajes de programación. Permite colorear la sintaxis del código, la navegación entre ficheros se hace mediante pestañas, facilita el manejo de código mediante el uso de sangrado y soporta extensiones.

4.11.3 SoapUI

SoapUI es una aplicación gratuita de código abierto que nos ha permitido hacer trabajos de test sobre los servicios proporcionados por la máquina virtual.

4.11.4 Agent Ransack

Potente aplicación de búsqueda de ficheros. Los puntos fuertes de esta utilidad son la rapidez de la búsqueda, su efectividad, las opciones disponibles y la forma de visualización.

5. Implementación

5.1 Definición de la arquitectura

Como ya se ha mencionado con anterioridad, la arquitectura del sistema implementado consiste en un servidor que proveerá servicios a otros sistemas a través de sus puertos. En la figura 16, se describe a grandes rasgos qué es la máquina virtual. La estructura interna y sus componentes se describen en la figura 17.

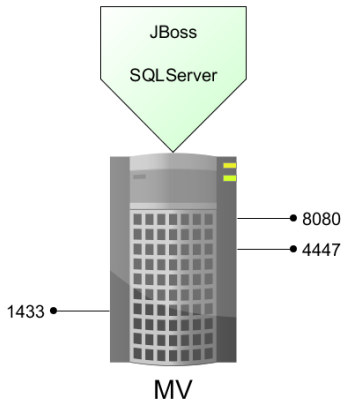


fig. 16

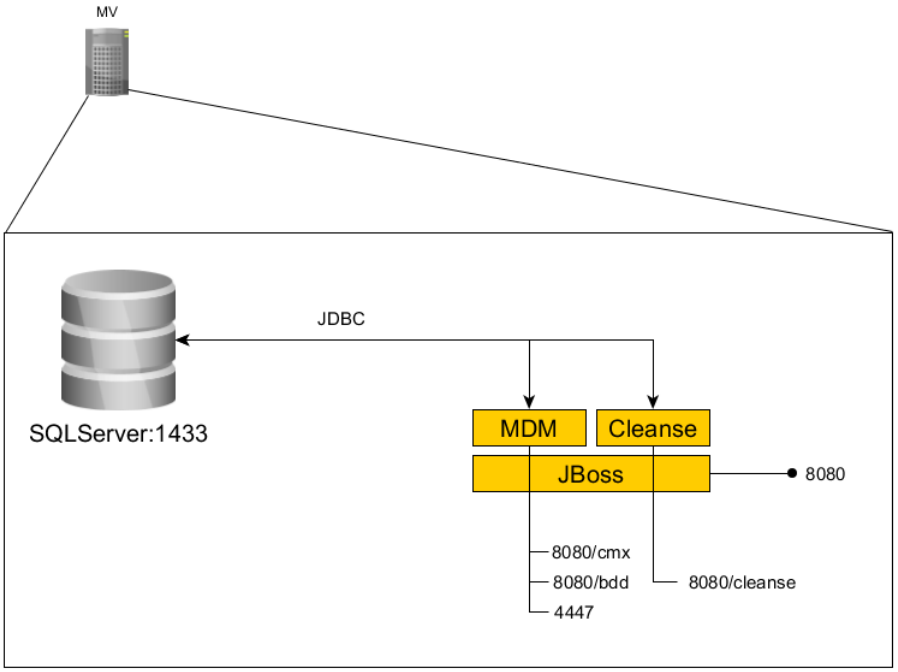


fig. 17

- *Hub MDM* (.ear): es fichero de despliegue de la aplicación que contiene el Hub MDM.
- *Cleanse Server* (.ear): es fichero de despliegue de la aplicación encargada de ejecutar los trabajos de procesamiento (funciones que reciben unas entradas que dan como resultados unas salidas, con un comportamiento transparente al desarrollador) y es usada, por el Hub MDM en las tareas de procesado de datos.
- *JBoss*: es un servidor de aplicaciones Java EE de código abierto que ofrece una plataforma de alto rendimiento para aplicaciones de e-business. Al estar basado en Java, JBoss puede ser utilizado en cualquier sistema operativo para el que esté disponible la máquina virtual de Java, en este caso, Windows Server (aunque también se ha utilizado con CentOS en las primeras fases del trabajo). JBoss se encarga de desplegar las aplicaciones de Hub MDM y Cleanse Server.
- *JDBC*: es una API que permite la ejecución de operaciones sobre bases de datos desde el lenguaje de programación Java, independientemente del sistema operativo donde se ejecute o de la base de datos a la cual se accede, utilizando el dialecto SQL del modelo de base de datos que se utilice. Mediante JDBC, los servicios del sistema 'atacan' a la base de datos.

Los puertos que se ponen, pues, a disposición de los sistemas cliente son los siguientes:

- *8080*: Puerto estándar HTTP del servidor de aplicaciones JBoss.
- *4447*: Puerto de interacción remota.
- *1433*: Puerto estándar de Microsoft SQL Server. Permite hacer acceder, consultar y modificar los datos que ha ido almacenando el sistema.

Como ya hemos avanzado en el apartado 4.3.2, se hace uso de NAT para traducir los puertos. En la siguiente tabla se muestra la equivalencia:

Máquina Virtual	Red Interna
10.0.2.15 : 8080	192.168.1.101 : 18080
10.0.2.15 : 4447	192.168.1.101 : 14447
10.0.2.15 : 1433	192.168.1.101 : 11433

En la figura 18 se muestra cómo quedaría finalmente la estructura del sistema.

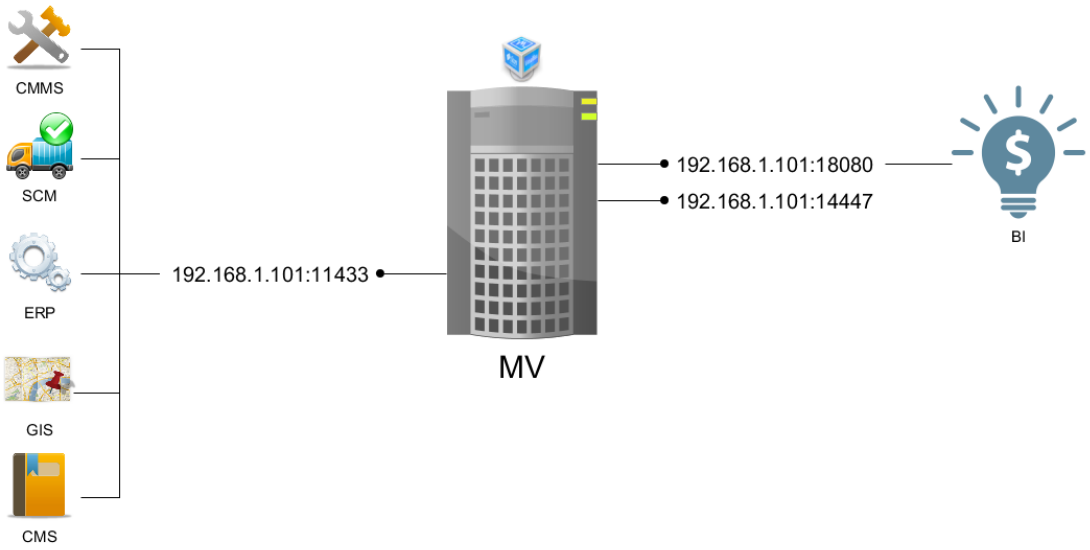


fig. 18

5.2 Sistema MDM

Para su desarrollo, es indispensable la colaboración de nuestro cliente ya que, volviendo a insistir, se deberá crear todo el sistema en base a los datos que considere críticos para su empresa. Todo el desarrollo se llevará acabo, como ya se ha explicado en apartados anteriores, desde la máquina virtual utilizando, principalmente, la Hub Console de MDM.

5.2.1 Creación de un Operational Record Store

Un 'Operational Record Store' o, en español, Almacén de Registros Operacional (ORS de ahora en adelante) es la base de datos que habrá que crear inicialmente y donde se almacenarán todos los datos relativos a la implementación del sistema MDM.

El ORS se creará desde línea de comandos, con la utilidad que provee la herramienta. En la siguiente tabla se muestran los valores con los que se creó nuestro ORS:

FIELD	VALUE
DB Type	MMSQL
ORS DB hostname	localhost
ORS DB port number	1433
ORS DB name	ANY
ORS DB path	C:\MSSQLDATA
ORS DB collation name	Latin1_General_CI_AS
DBA username	sa

Una vez creado, se añadirá en la Hub Console en el apartado 'Databases'. En ese momento, podremos empezar el desarrollo.

5.2.2 Diseño del Tarjet Data Model

A continuación, se mostrará el diseño lógico que se consultará a la hora de crear los Base Objects, según el esquema diseñado en base a los requerimientos acordados con el cliente (figura 19).

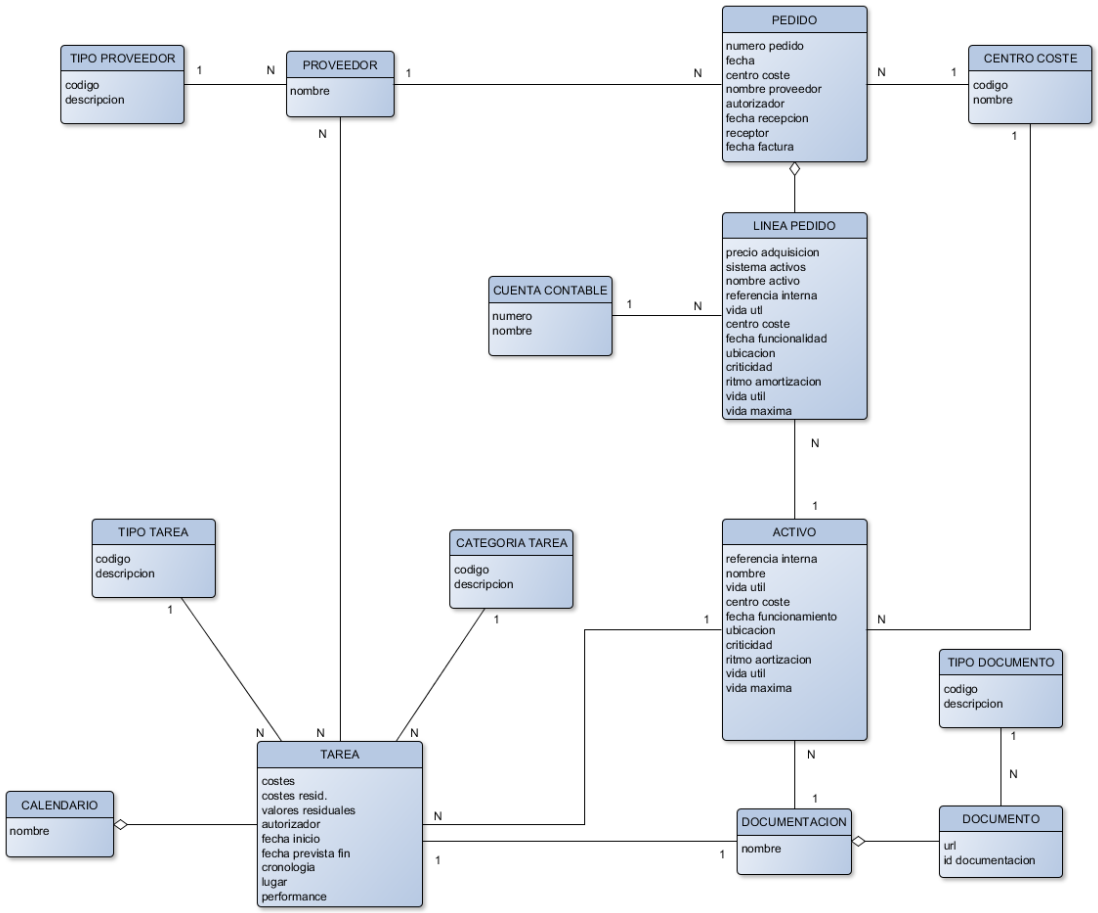


fig. 19

Se debe de tener en cuenta que los valores de las claves principales tendrán tipo char(14), las fechas serán de tipo 'date' y las tiras de tipo 'varchar'. El campo 'Rowid_object' lo inserta el sistema MDM automáticamente al crear la tabla, y será su clave primaria. La definición se realizará en inglés para maximizar la compatibilidad.

C_ORDER (Rowid_object: char(14),

order number: varchar(50),

order date: date,

cost_center: varchar(50),

provider_id: char(14),

provider_name: varchar(50),

authorizer: varchar(50),

receiver: varchar(50),

invoice_date: date,

receipt_date: date,

cost_center_id: char(14))

CP: {Rowid_object}

CAj: {provider_id} -> C_PROVIDER

CAj: {cost_center_id} -> C_COST_CENTER

C_ASSET (Rowid_object: char(14),

name: varchar(50)

useful_life: varchar(50)

cost_center: varchar(50)

start_up_date: date

location: varchar(50)

criticality: varchar(50)

amortization_rate: varchar(50)

maximum_life: varchar(50)

documentatio_id: char(14)

cost_center_id: char(14)

order_line_id: char(14)

internal_reference: varchar(50))

CP: {Rowid_object}

CAj: {documentation_id } -> C_DOCUMENTATION

CAj: {order_line_id } -> C_ORDER_LINE

CAj: {cost_center_id } -> C_COST_CENTER

C_CALENDAR (Rowid_object: char(14),

name: varchar(50))

CP: {Rowid_object}

C_COST_CENTER (Rowid_object: char(14),
code: varchar(50),
name: varchar(50))

CP: {Rowid_object}

C_DOCUMENT (Rowid_object: char(14),
document_type: varchar(50),
url: varchar(50),
documentation_id: char(14),
source_system: varchar(50))

CP: {Rowid_object}
CAj: {document_type } -> C_DOCUMENT_TYPE
CAj: {documentation_id } -> C_DOCUMENTATION

C_DOCUMENT_TYPE (Rowid_object: char(14),
code: varchar(50),
label: varchar(50))

CP: {Rowid_object}
Unico: {code}

C_DOCUMENTATION (Rowid_object: char(14),
name: varchar(50))

CP: {Rowid_object}

C_LEDGER_ACOUNT (Rowid_object: char(14),
Ldgnumber: varchar(50),
name: varchar(50))

CP: {Rowid_object}

C_ORDER_LINE (Rowid_object: char(14),
purchase_price: varchar(50),
internal_reference: varchar(50),
account_id: char(14),
order_id: char(14),
asset_name: varchar(50),
useful_life: varchar(50),
cost_center: varchar(50),
start_up_date: date,
location: varchar(50),
criticality: varchar(50),
amortization_rate: varchar(50),
máximum_life: varchar(50),
asset_system: varchar(50))

CP: {Rowid_object}
CAj: {account_id } -> C_LEDGER_ACCOUNT
CAj: {order_id } -> C_ORDER

C_PROVIDER (Rowid_object: char(14),
provider_name: varchar(50),

type: varchar(50))

CP: {Rowid_object}

CAj: {type } -> C_PROVIDER_TYPE

C_PROVIDER_TYPE (Rowid_object: char(14),

code: varchar(50),

label: varchar(50))

CP: {Rowid_object}

Unico: {code}

C_TASK (Rowid_object: char(14),

type: varchar(50),

category: varchar(50),

costs: varchar(50),

waste_costs: varchar(50),

residual_value: varchar(50),

authorizer: varchar(50),

place: varchar(50),

chronology: varchar(50),

start_date: date,

name: varchar(50),

expected_end_date: date,

calendar_id: char(14),

documentation_id: char(14),

asset_id: char(14),

performance: varchar(50))

CP: {Rowid_object}

CAj: {calendar_id } -> C_CALENDAR

CAj: {category } -> C_TASK_CATEGORY

CAj: {asset_id } -> C_ASSET

CAj: {type } -> C_TASK_TYPE

CAj: {documentation_id } -> C_DOCUMENTATION

C_TASK_CATEGORY (Rowid_object: char(14),

code: varchar(50),

label: varchar(50))

CP: {Rowid_object}

Unico: {code}

C_TASK_TYPE (Rowid_object: char(14),

code: varchar(50),

label: varchar(50))

CP: {Rowid_object}

Unico: {code}

C_REL_TASK_PROVIDER (Rowid_object: char(14),

task_id: char(14),

provider_id: char(14))

CP: {Rowid_object}

CAj: {provider_id } -> C_PROVIDER

CAj: {task_id } -> C_TASK

5.2.3 Implementación de los Base Objects

Las tablas de los Base Objects (las que proveerán los 'Golden Records' que servirán como punto de referencia común para consultar la información consolidada de la organización) se crearán en el apartado 'Schema' de la Hub Console, seleccionando el ORS creado. Una vez creadas, podremos consultar el esquema en el apartado 'Schema Viewer'.

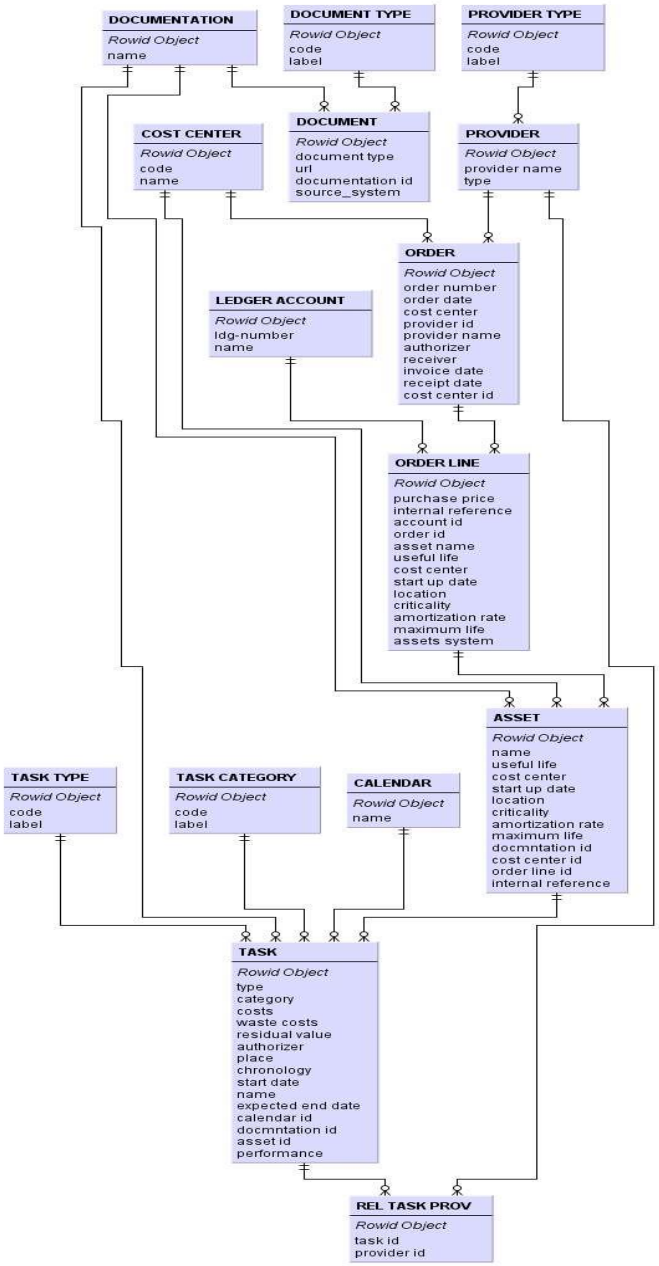


fig. 20 Esquema resultante tras la creación de las tablas Base Object

5.2.4 Definición de Landings

Las tablas Landing son iguales que las tablas Base Objects (excepto las listas de valores, que se juntarán en una sola tabla), pero con dos campos extra, 'source_system' (para las tablas en los que varias fuentes proveen datos) y 'source_id', que indicarán el sistema fuente del que provee ese registro y cuál era su identificador en dicho sistema.

C_L_ORDER (order number: varchar(50),

order date: date,

cost_center: varchar(50),

provider_id: char(14),

provider_name: varchar(50),

authorizer: varchar(50),

receiver: varchar(50),

invoice_date: date,

receipt_date: date,

cost_center_id: char(14)

source_id: varchar(255),

source_system: varchar(50))

C_L_ASSET (name: varchar(50)

useful_life: varchar(50)

cost_center: varchar(50)

start_up_date: date

location: varchar(50)

criticality: varchar(50)

amortization_rate: varchar(50)

maximum_life: varchar(50)

documentatio_id: char(14)

cost_center_id: char(14)

order_line_id: char(14)

internal_reference: varchar(50)

source_id: varchar(255),

source_system: varchar(50))

C_L_CALENDAR (name: varchar(50)

source_id: varchar(255))

C_L_COST_CENTER (code: varchar(50),

name: varchar(50)

source_id: varchar(255))

C_L_DOCUMENT (document_type: varchar(50),

url: varchar(50),

documentation_id: char(14),

source_system: varchar(50)

source_id: varchar(255))

C_L_DOCUMENTATION (name: varchar(50),

source_id: varchar(255))

C_L_LEDGER_ACOUNT (Ldgnumber: varchar(50),

name: varchar(50),

source_id: varchar(255))

C_L_ORDER_LINE (purchase_price: varchar(50),

internal_reference: varchar(50),

account_id: char(14),

order_id: char(14),

asset_name: varchar(50),

useful_life: varchar(50),

cost_center: varchar(50),

start_up_date: date,

location: varchar(50),

criticality: varchar(50),

amortization_rate: varchar(50),

máximum_life: varchar(50),

asset_system: varchar(50),

source_id: varchar(255),

source_system: varchar(50))

C_L_PROVIDER (provider_name: varchar(50),

type: varchar(50),

source_id: varchar(255),

source_system: varchar(50))

C_L_TASK (type: varchar(50),

category: varchar(50),

costs: varchar(50),

waste_costs: varchar(50),

residual_value: varchar(50),

authorizer: varchar(50),

place: varchar(50),

chronology: varchar(50),

start_date: date,

name: varchar(50),

expected_end_date: date,

calendar_id: char(14),

documentation_id: char(14),

asset_id: char(14),

performance: varchar(50),

source_id: varchar(255),

source_system: varchar(50))

Al crear cada tabla, el sistema añade automáticamente el campo Last_Update_Date, el cual sirve para identificar la fecha de la última actualización del registro que se hizo en la fuente.

5.2.5 Listas de valores

El cliente define las tablas 'TASK_CATEGORY', 'TASK_TYPE', 'PROVIDER_TYPE' y 'DOCUMENT_TYPE' como listas de valores (LOV de ahora en adelante), ya que no son datos insertados por usuarios de los sistemas fuentes, sino una lista de datos predefinidos que se muestran en los formularios. La peculiaridad que tiene esta tabla es que se le deberá indicar su destino a través del campo 'tarjet'.

C_L_LOV (label: varchar(50),

code: varchar(50),

source_id: varchar(255),

source_system: varchar(50)

tarjet: varchar(50))

5.2.6 Definición de Staggings

Las tablas se crean dentro de las especificaciones de cada tabla Base Object, como si fueran una propiedad (aunque, realmente, sean tablas independientes). Al crear cada tabla Staging, automáticamente podemos elegir los campos de esa tabla Base Object que contiene. El número de tabla Staging de cada tabla Base Object depende del número de fuentes que provean datos a dicha tabla. Para determinar esto último, hemos hecho un estudio previo junto al cliente.

A continuación, se detallan los campos incluidos en cada tabla Staging, dependiendo de la tabla Base Object que corresponda. En el propio nombre de la tabla Staging se especifica el sistema que provee sus datos.

C_ASSET:

- C_S_CMMS_ASSET {name, useful life, criticality, internal_reference}
- C_S_GIS_ASSET {name, location, internal_reference}
- C_S_SCM_ASSET {name, start_up_date, internal_reference}

C_COST_CENTER:

- C_S_ERP_COSTCENT {code, name}

C_DOCUMENT:

- C_S_CMS_DOC {document_type, url}
- C_S_CMMS_DOC{document_type, url, documentation_id}

C_DOCUMENTATION:

- C_S_CMS_DOCUMNTION {name}

C_LEDGER_ACCOUNT:

- C_S_ERP_LEDGACC: {name}

C_ORDER:

- C_S_ERP_ORDER {order_number, order_date, cost_center, provider_id, provider_name, authorizer, invoice_date}
- C_S_SCM_ORDER {order_number, receiver, receipt_date}

C_ORDER_LINE:

- C_S_ERP_ORDER_LINE {purchase_price, order_id, asset_name, amortization_rate, assets_system}
- C_S_SCM_ORDER_LINE {internal_reference, order_id}

C_PROVIDER:

- C_S_CMMS_PROVIDER {provider_name, type}
- C_S_ERP_PROVIDER {provider_name, type }
- C_S_SCM_TASK {provider_name, type }

C_TASK:

- C_S_CMMS_TASK {type, category, costs, start_date, name, calendar_id, asset_id, performance}
- C_S_ERP_TASK {type, category, start_date, name, calendar_id, docmntation_id, asset_id}
- C_S_SCM_TASK {type, costs, authorizer, start_date, name, expected_end_date, asset_id, performance}

C_CALENDAR:

- C_S_CMMS_CALENDAR {name}

C_DOCUMENT_TYPE:

- C_S_CMS_DOC_T_LOV {code, label}
- C_S_CMMS_DOC_T_LOV { code, label}

C_PROVIDER_TYPE:

- C_S_SCM_PROV_T_LOV {code, label}
- C_S_CMMS_PROV_T_LOV {code, label}
- C_S_ERP_PROV_T_LOV {code, label}

C_TASK_CATEGORY:

- C_S_CMMS_CATEG_T_LOV {code, label}

C_TASK_TYPE:

- C_S_SCM_TASK_T_LOV {code, label}
- C_S_CMMS_TASK_T_LOV {code, label}

5.2.7 Definición de Mappings

Dentro del apartado 'Mappings' y mediante una interfaz gráfica, especificamos las relaciones entre tablas Landing y tablas Staging. Si señalamos la opción 'Enable Condition' en el apartado 'Query Parameters', podremos añadir condiciones (similar a la cláusula WHERE de SQL sobre la tabla Landing) para cuando se ejecute el Mapping correspondiente.

En muchas ocasiones será necesario definir una serie de funciones para manipular los datos de origen para cargarlos en las tablas Staging. Estas funciones se definen dentro del apartado 'Cleanse Functions'. A continuación, se muestran las funciones definidas para utilizarlas en los Mappings:

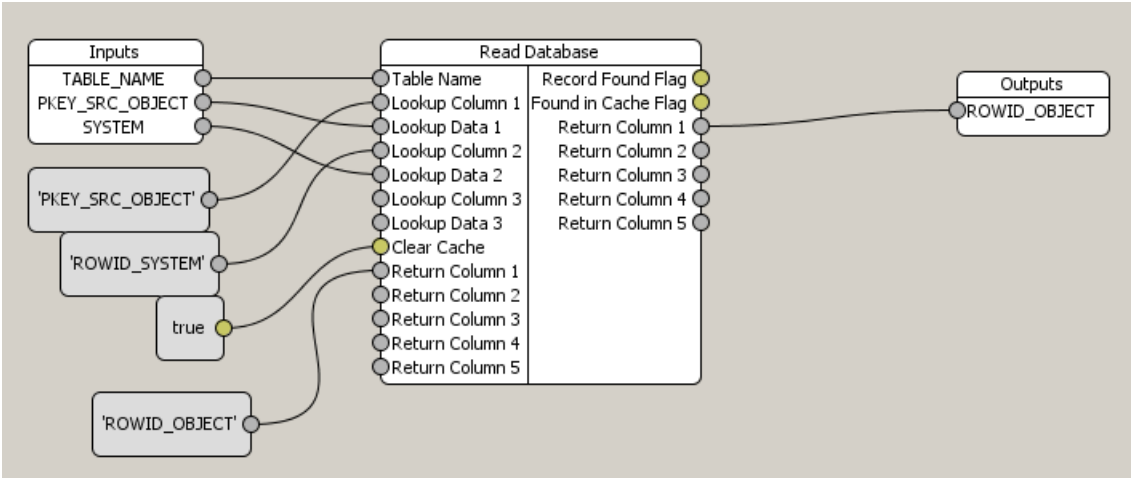


fig. 21 LOOKUP_ROWID

Esta función permite insertar un nombre de una tabla y valores en los campos 'PKEY_SRC_OBJECT' y 'ROWID_SYSTEM', retornando el valor de la columna 'ROWID_OBJECT'.

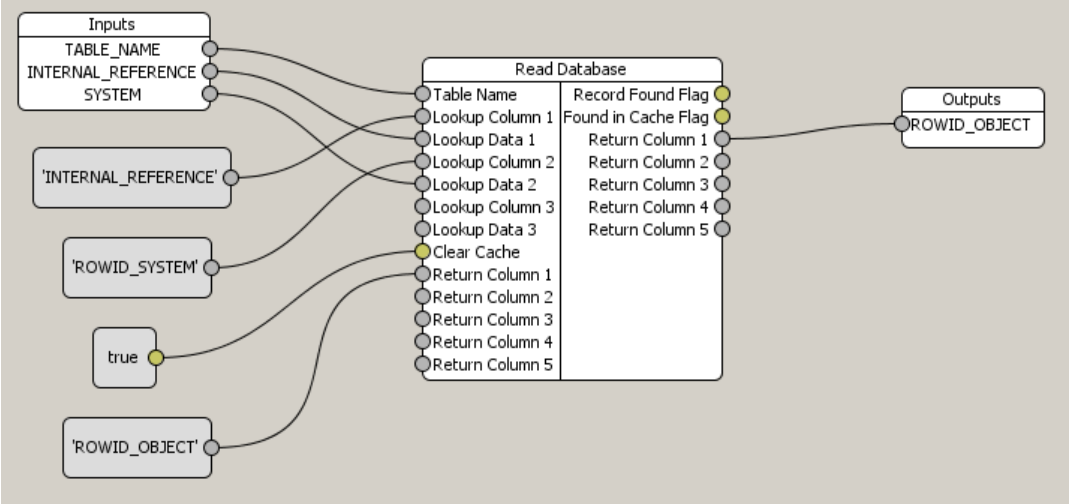


fig. 22 LOOKUP_ROWID_INT_REF

Esta función permite insertar un nombre de una tabla y valores en los campos 'INTERNAL_REFERENCE' y 'ROWID_SYSTEM', retornando el valor de la columna 'ROWID_OBJECT'.

Como sabemos qué tipo de tablas vamos a utilizar con estas funciones, sabemos con certeza que las columnas que hemos definido existirán en el momento en que se usen.

Estas funciones se utilizarán unas tablas de históricos de inserción en las tablas Base Object llamadas tablas 'XREF'. El objetivo es saber si se ha insertado el mismo registro con anterioridad, para así poder actualizarlo si fuera necesario.

Los Mappings definidos quedarían de la siguiente manera:

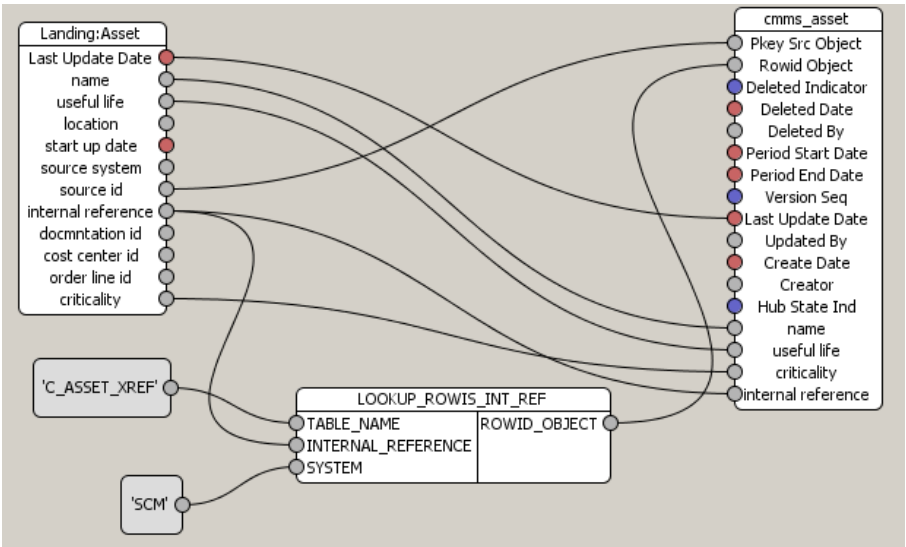


fig. 23 M_CMMS_ASSET

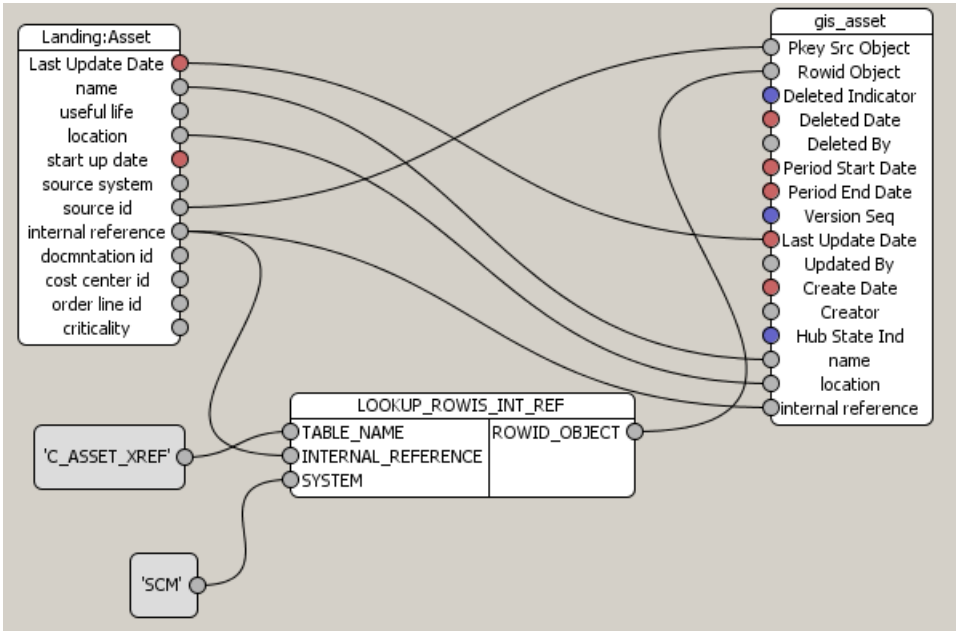


fig. 24 M_GIS_ASSET

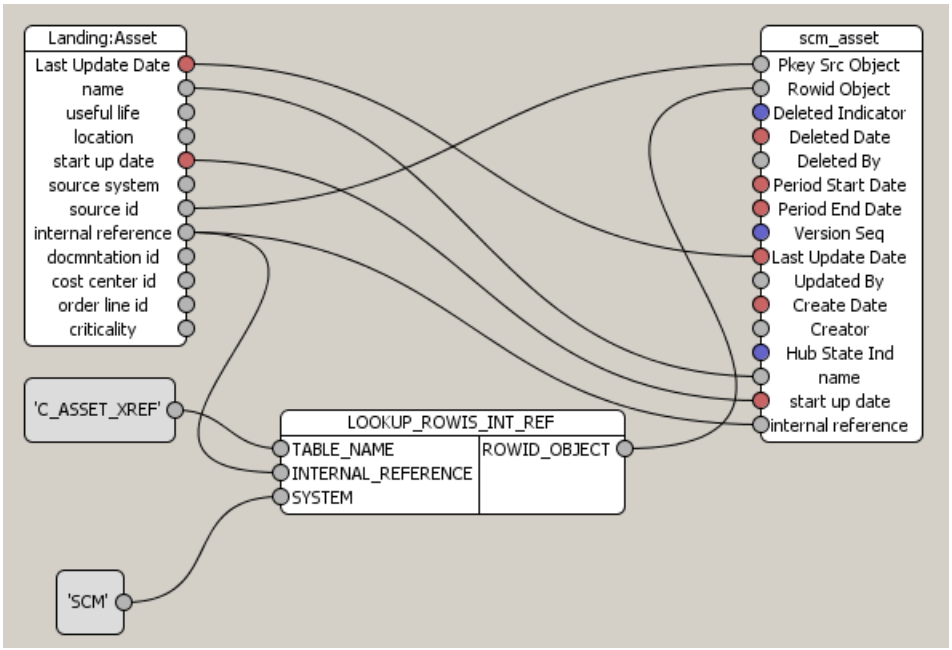


fig. 25 M_SCM_ASSET

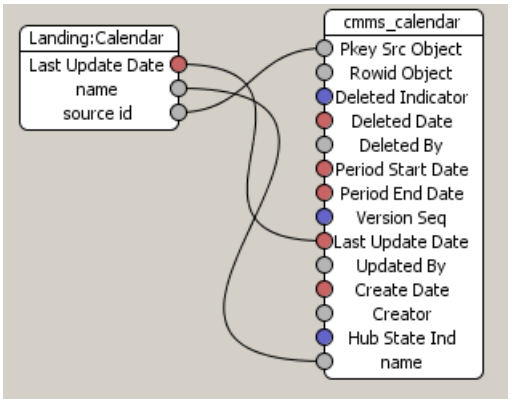


fig. 26 M_CMMS_CALENDAR

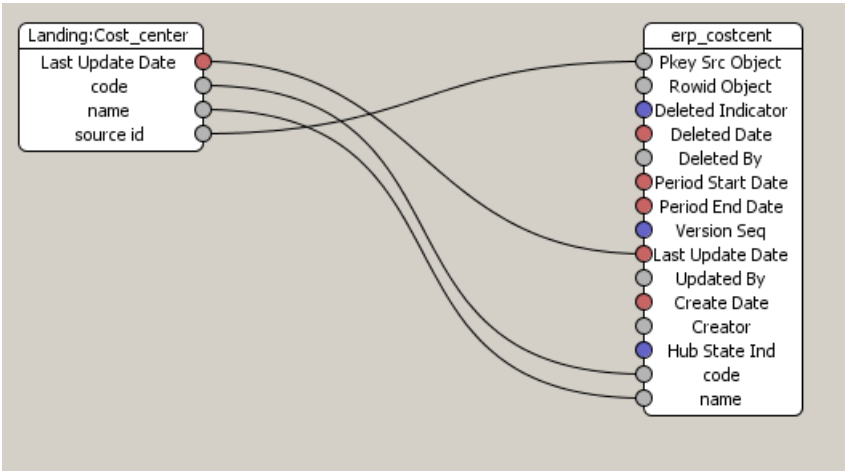


fig. 27 M_ERP_COSTCENT

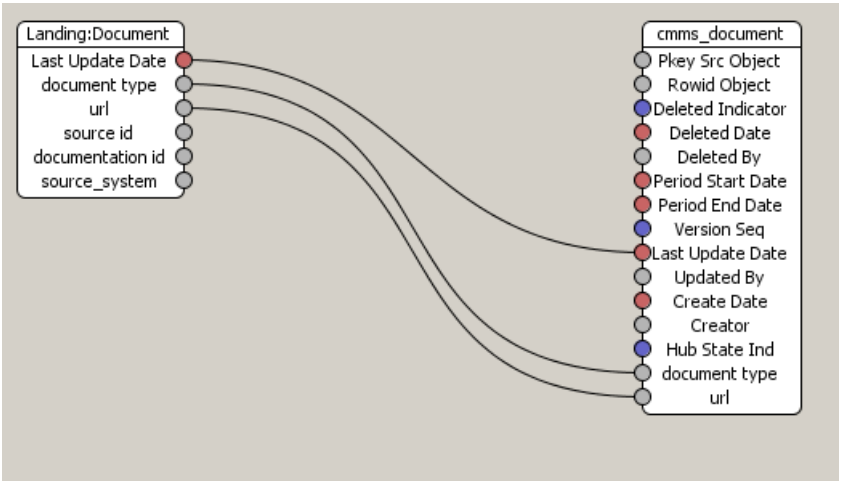


fig. 28 M_CMMS_DOC

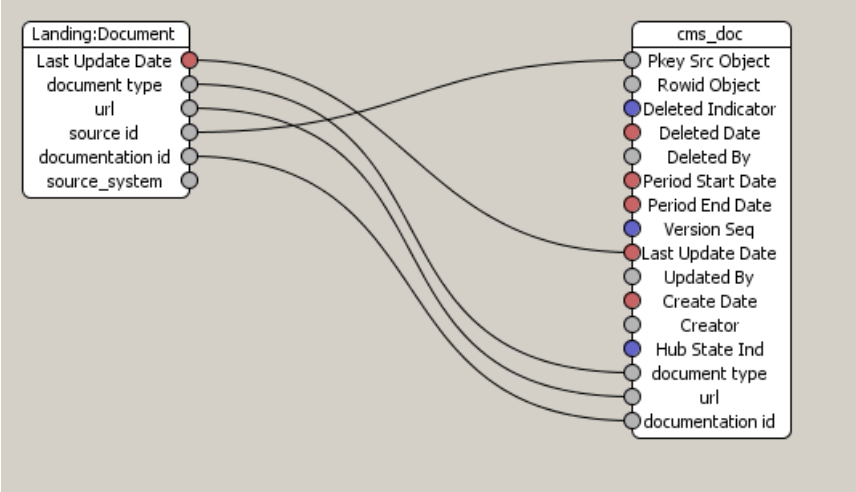


fig. 29 M_CMS_DOC

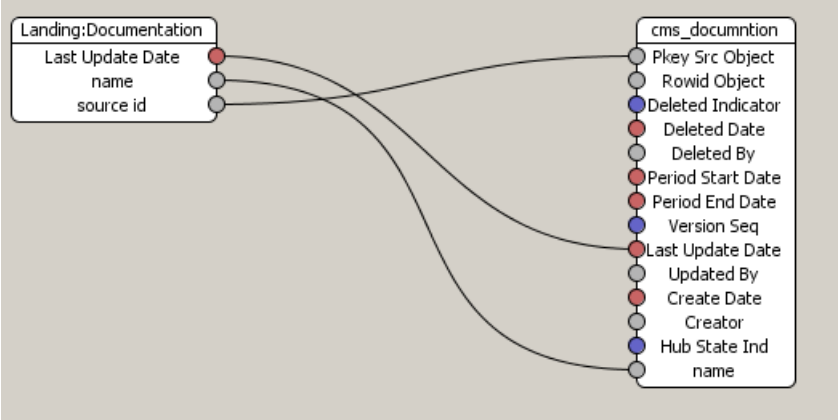


fig. 30 M_CMS_DOCUMNTION

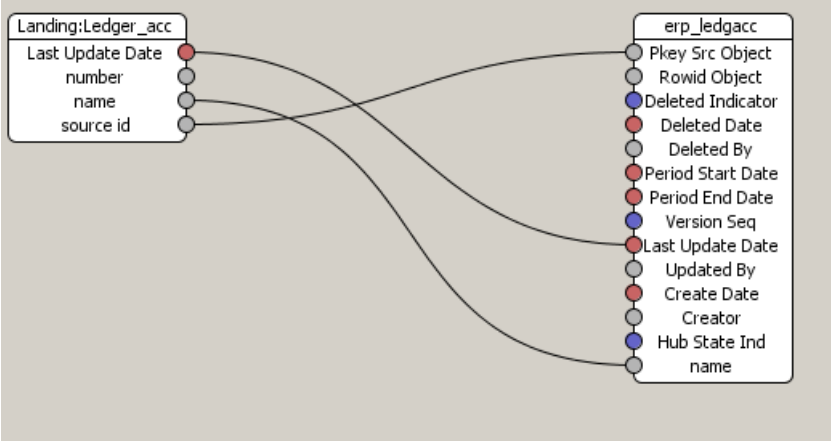


fig. 31 M_ERP_LEDACC

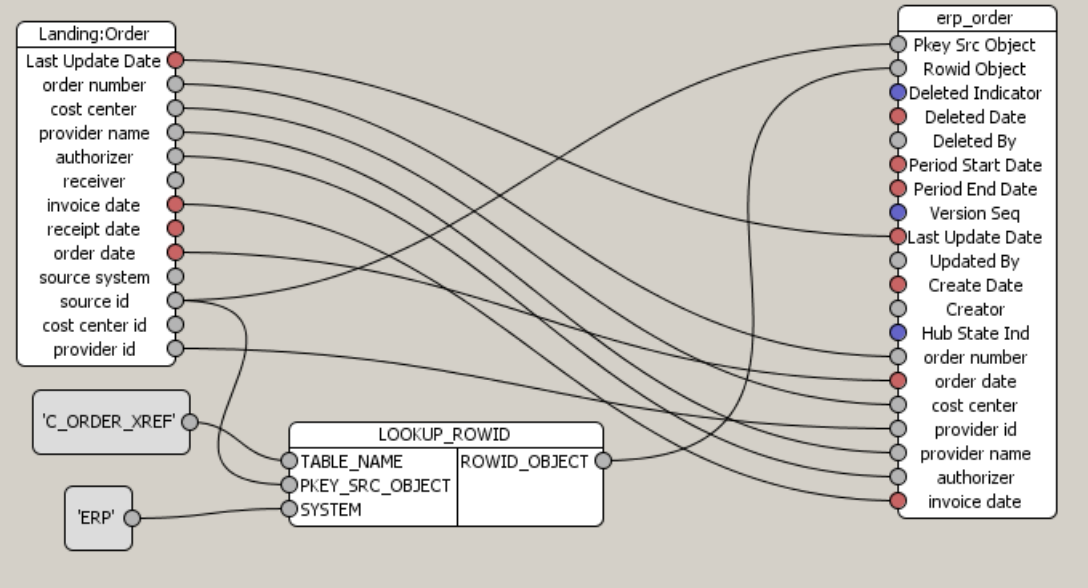


fig. 32 M_ERP_ORDER

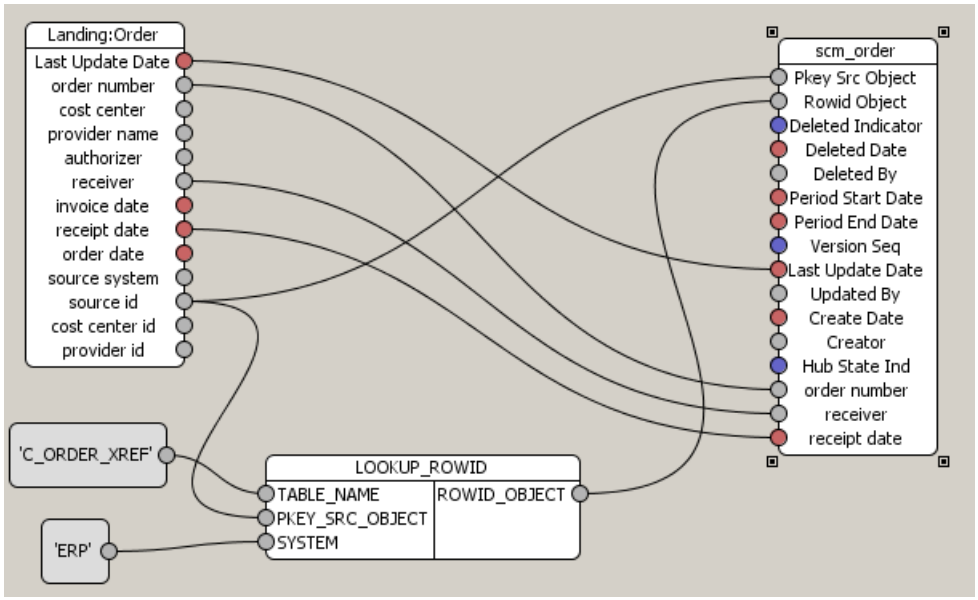


fig. 33 M_SCM_ORDER

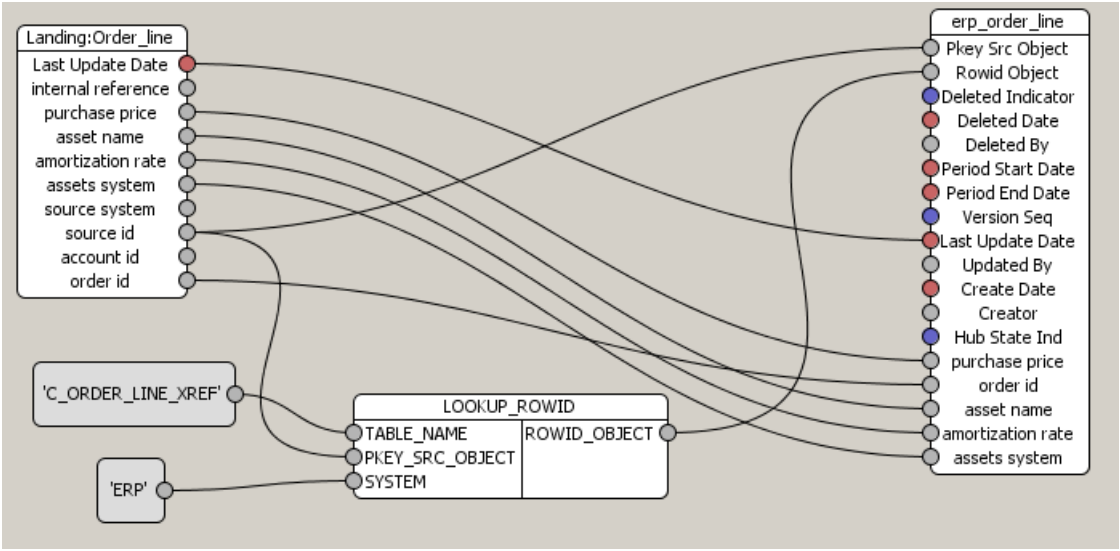


fig. 34 M_ERP_ORDER_LINE

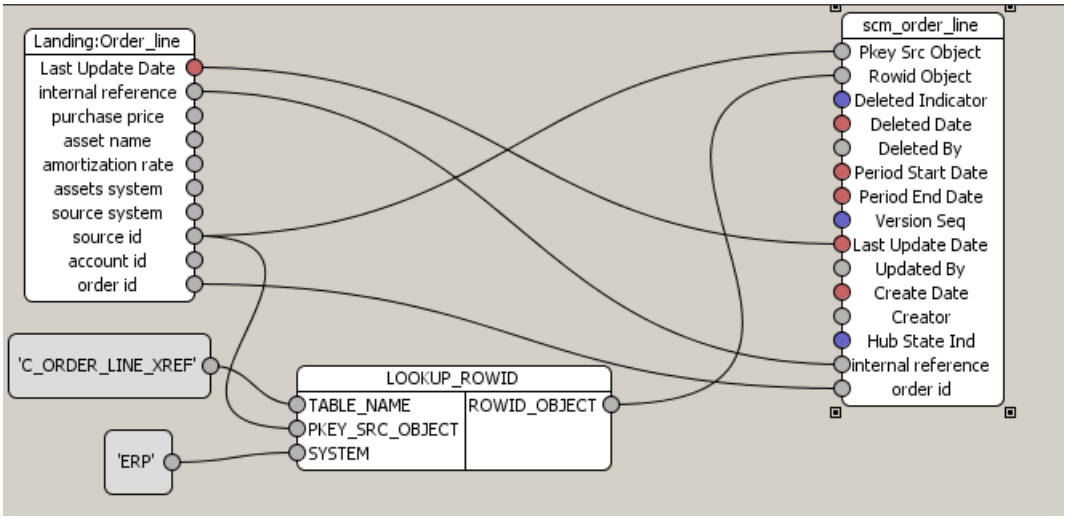


fig. 35 M_SCM_ORDER_LINE

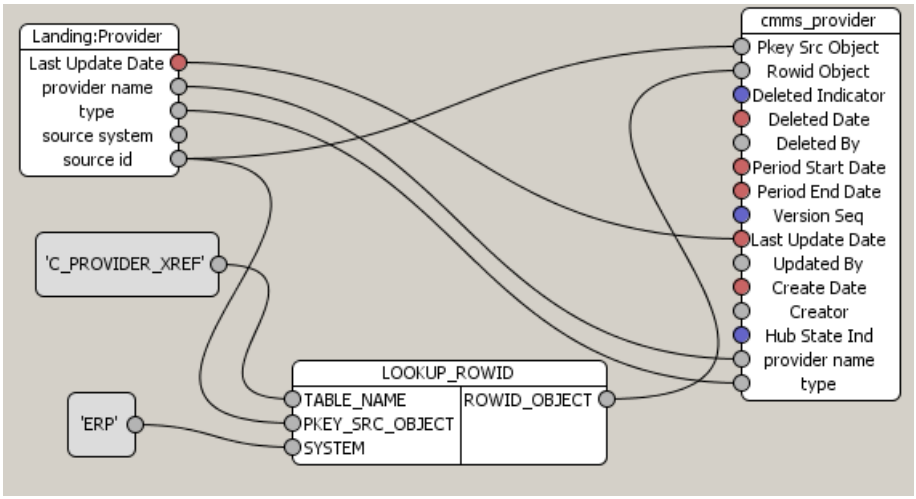


fig. 36 M_CMMS_PROVIDER

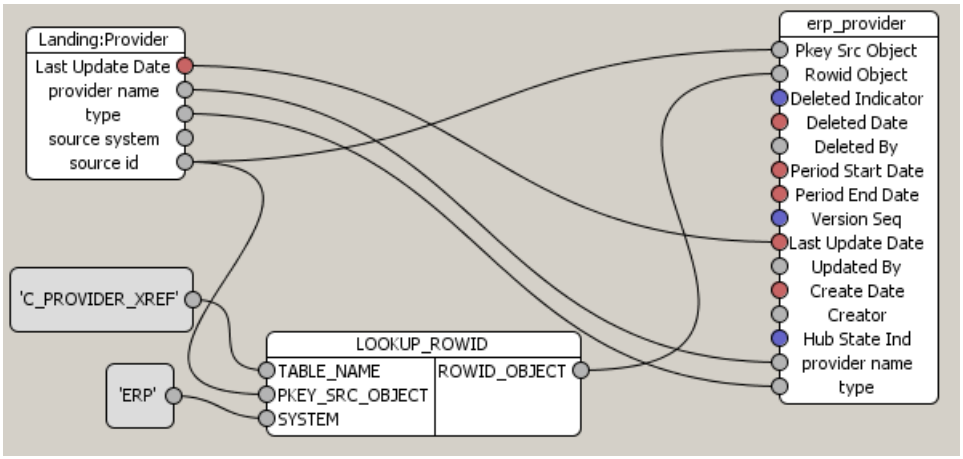


fig. 37 M_ERP_PROVIDER

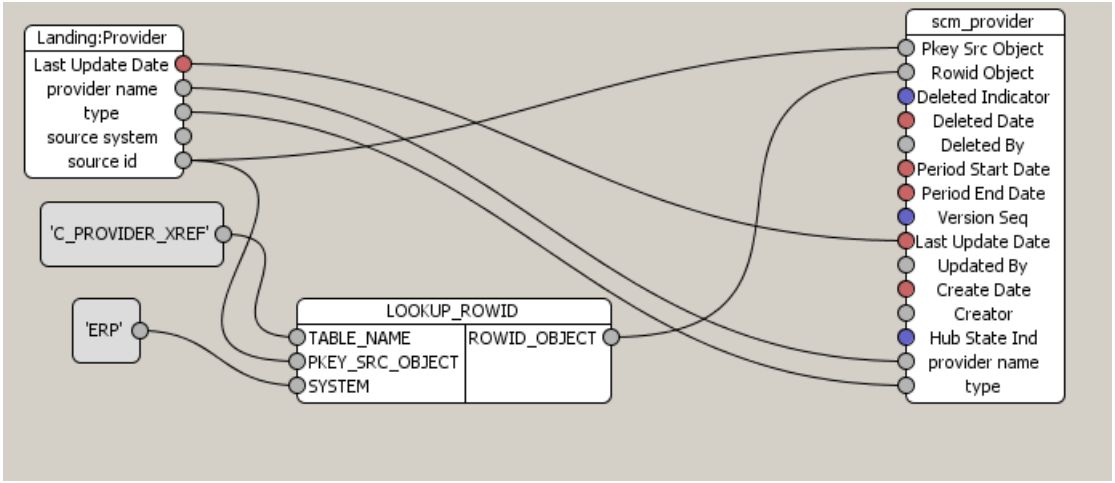


fig. 38 M_SCM_PROVIDER

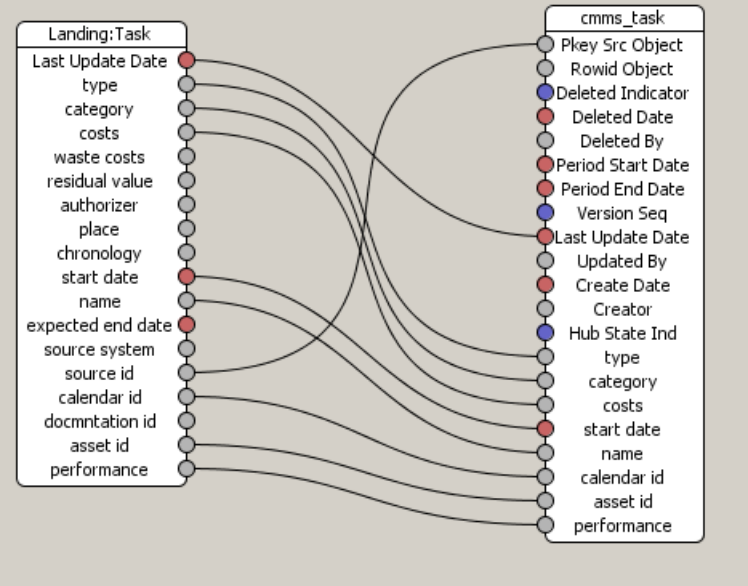


fig. 39 M_CMMS_TASK

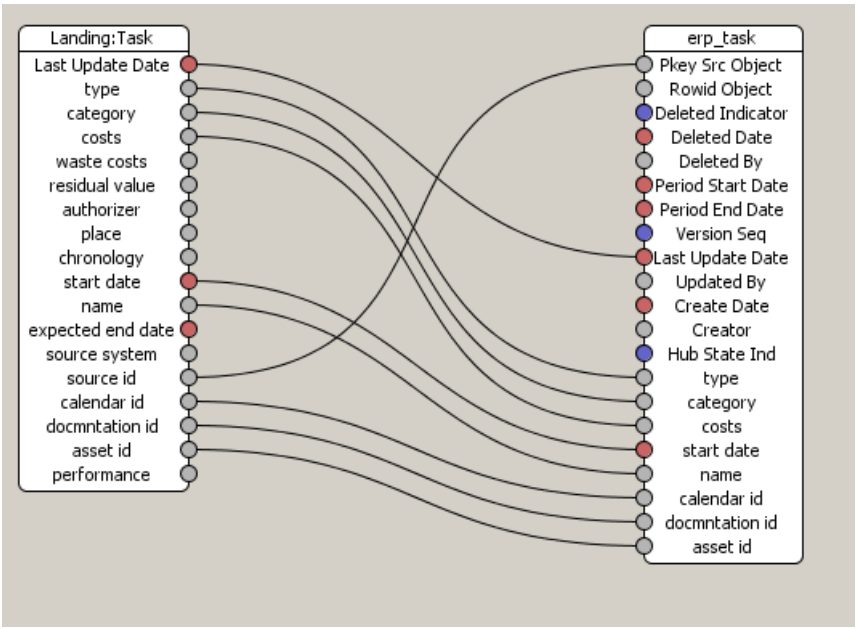


fig. 40 M_ERP_TASK

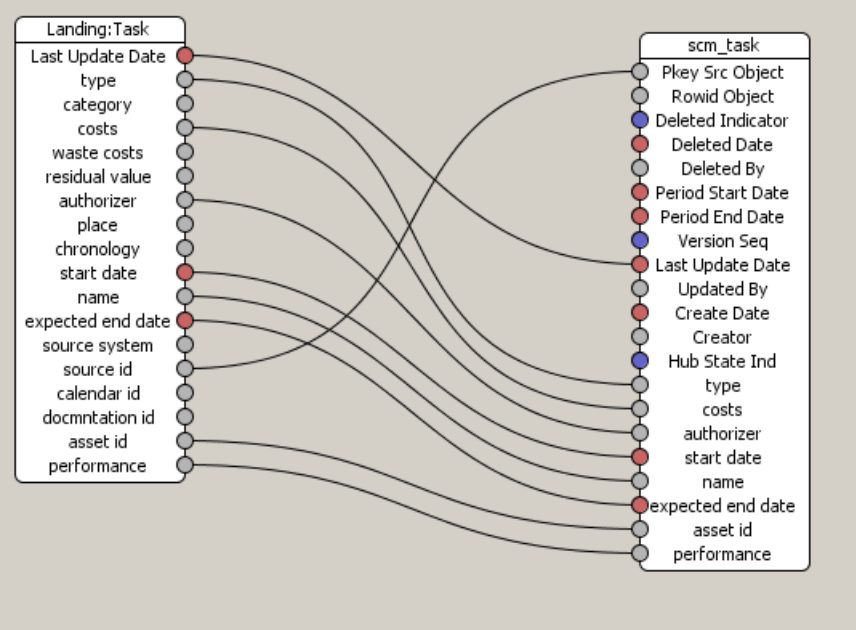


fig. 41 M_SCM_TASK

Los Mappings de las LOV se implementan de la misma forma en todos los casos. A continuación se muestra un ejemplo (figura 42).

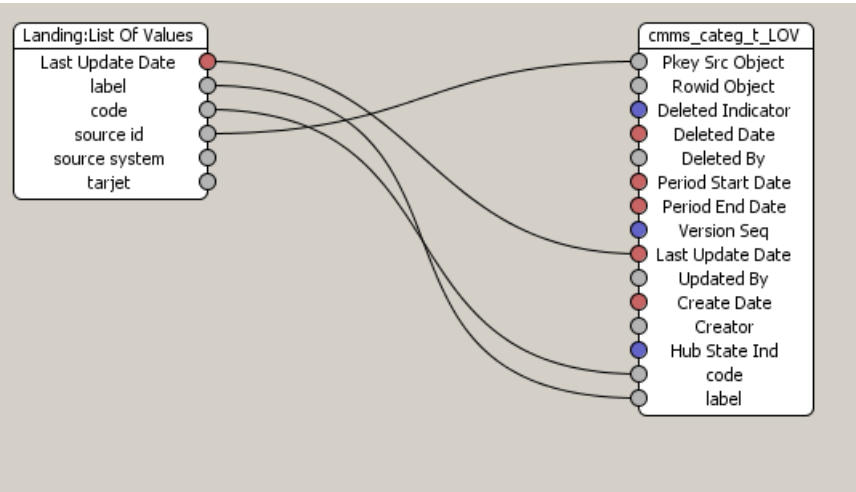


fig. 42 M_CMMS_CATEG_T_LOV

La peculiaridad de los Mappings de LOV es que se tienen que añadir como condición, aparte de la fuente en el valor 'SOURCE_SYSTEM', la tabla destino en el campo 'TARJET'. Como ejemplo se muestra el de M_ERP_PROV_T_LOV (figura 43).

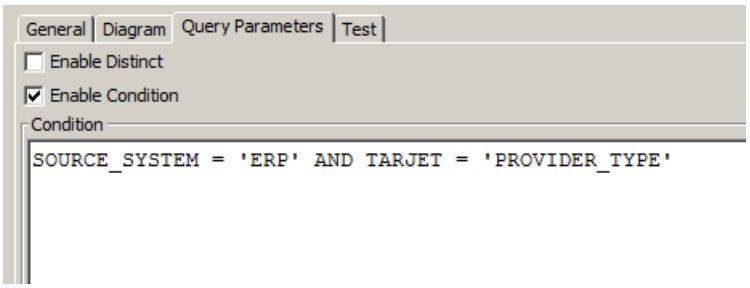


fig. 43 Condición de Mapping de LOV

Después de todo el proceso de desarrollo, nos queda una estructura como la que se muestra en la figura 44.

fig. 44

5.2.8 Creación de los Batch Groups

Es muy importante que se carguen los datos en el orden adecuado, sobre todo en la carga inicial. Para ello hemos definido la siguiente jerarquía para saber el orden según la dependencia entre tablas Base Objects antes de implementar los Batch Groups en la Hub

Console. Como se puede apreciar en la figura 45, se definirán, finalmente, 7 (uno para las LOV, que siempre deberán ir por separado).

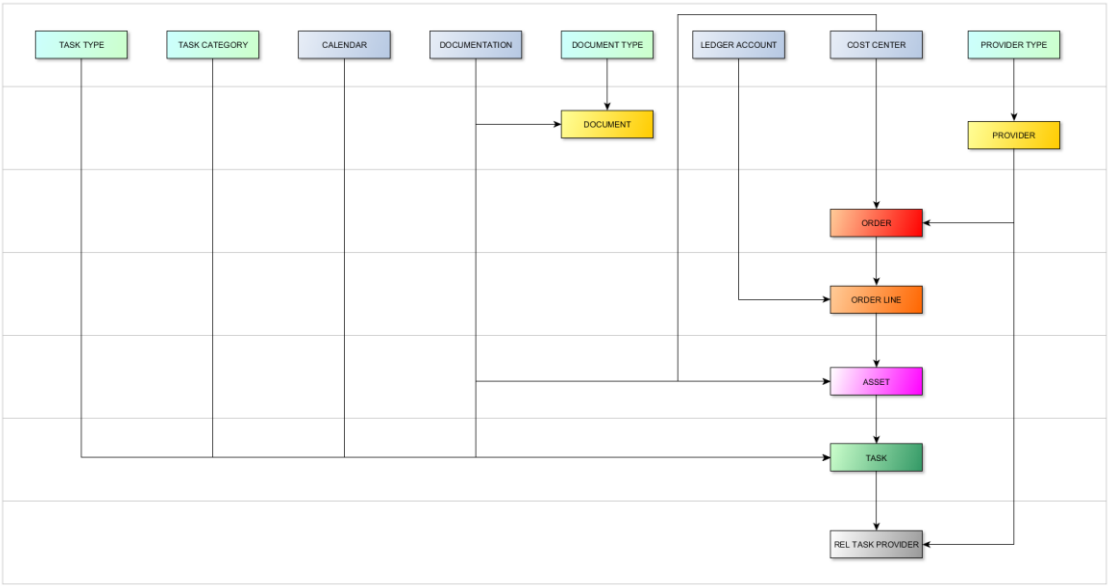


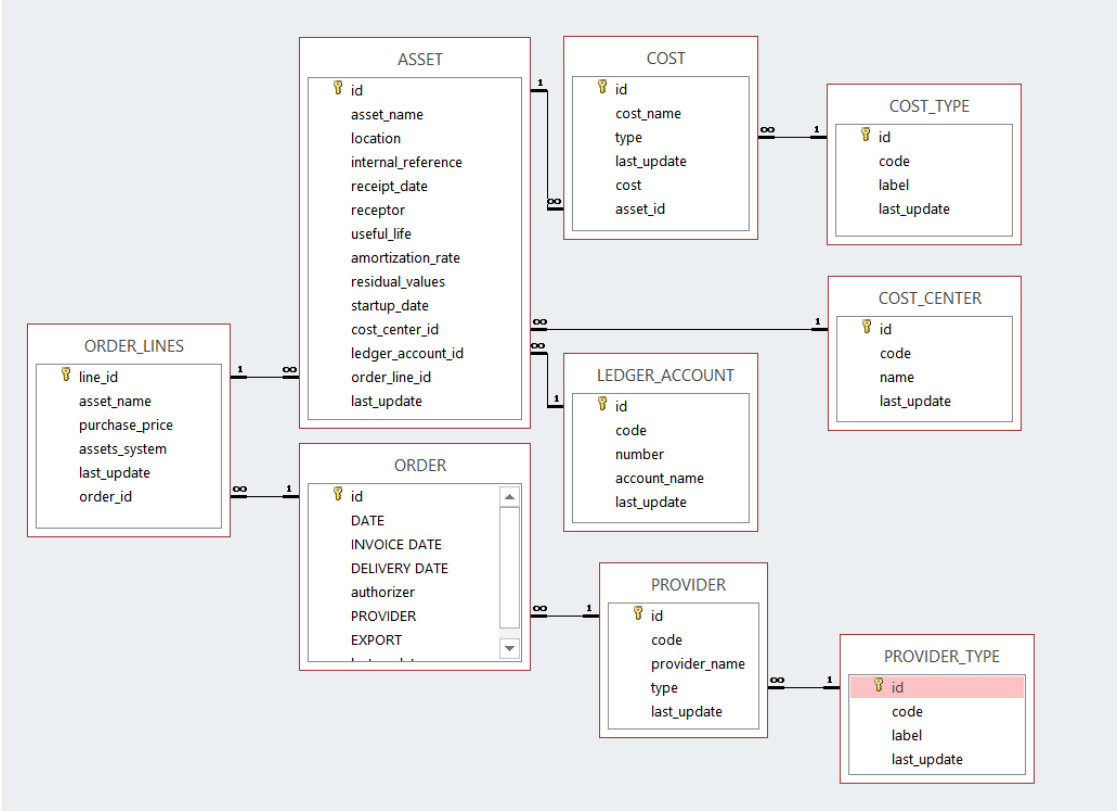
fig. 45 Esquema de Batch Groups

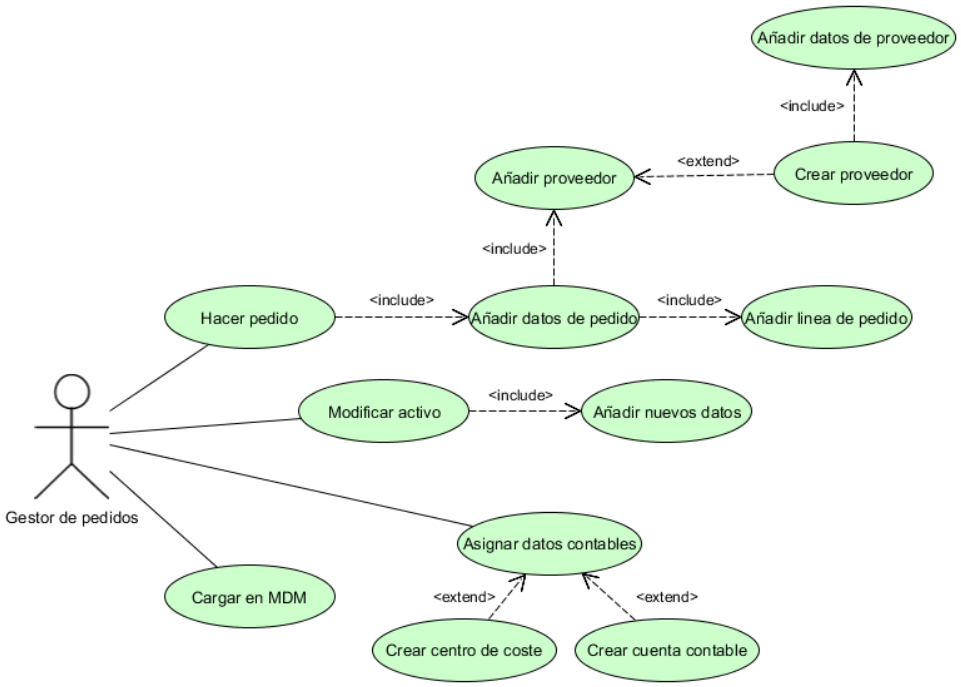
5.3 Simulación de las fuentes de datos

Para cada sistema fuente, se definirá su modelo de datos en base al caso de estudio, se crearán las tablas y relaciones correspondientes en Access y los formularios que, en definitiva, son los que se emplearán para simular el comportamiento de dichos sistemas (no es el objetivo de este trabajo ahondar en la implementación de estos sistemas). Todos los formularos se realizarán siguiendo el asistente de Access.

5.3.1 Sistema ERP

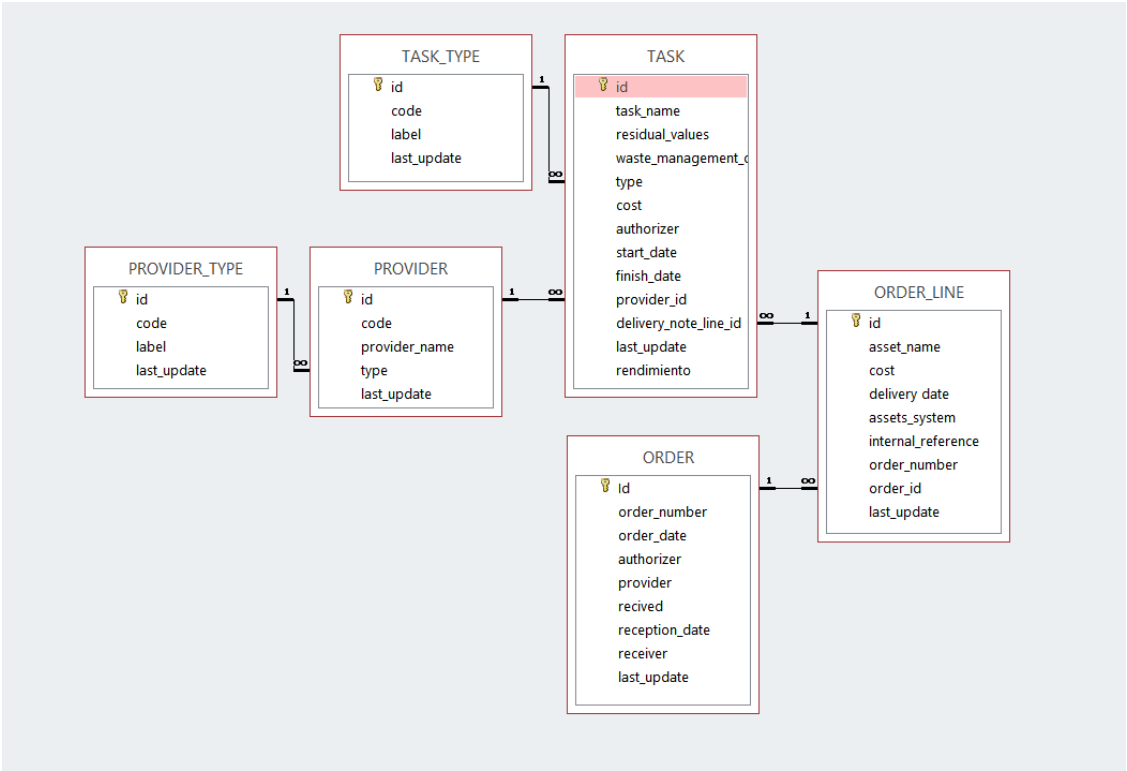
A continuación, se muestra la base de datos definida para el sistema ERP y los diagramas de casos de uso para la implementación de los formularios:

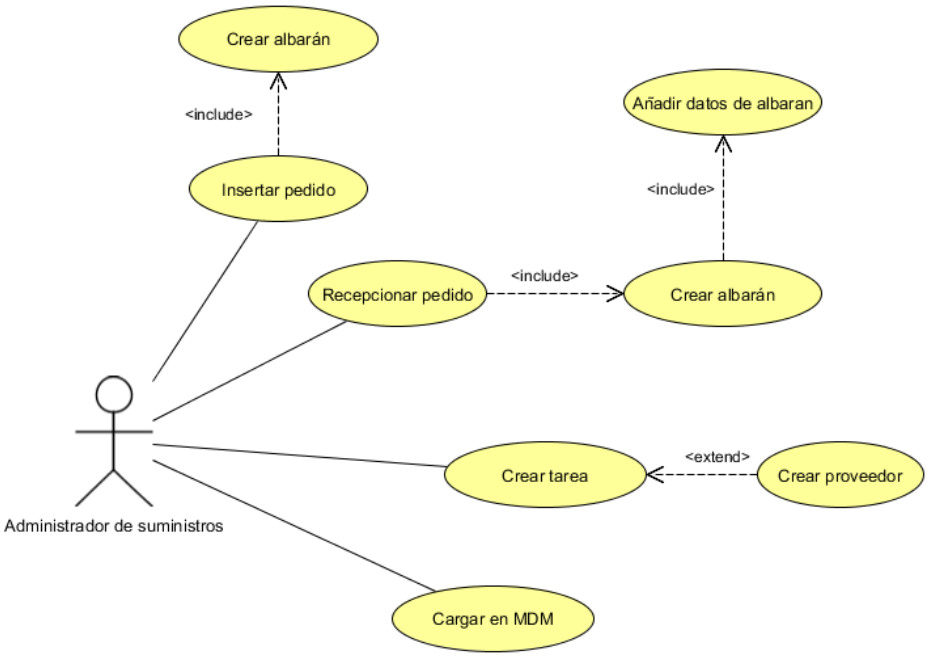

ASSET
id
asset_name
location
internal_reference
receipt_date
receptor
useful_life
amortization_rate
residual_values
startup_date
cost_center_id
ledger_account_id
order_line_id
last_update
COST
id
cost_name
type
last_update
cost
asset_id
COST_TYPE
id
code
label
last_update
COST_CENTER
id
code
name
last_update
LEDGER_ACCOUNT
id
code
number
account_name
last_update
ORDER_LINES
line_id
asset_name
purchase_price
assets_system
last_update
order_id
ORDER
id
DATE
INVOICE DATE
DELIVERY DATE
authorizer
PROVIDER
EXPORT
PROVIDER
id
code
provider_name
type
last_update
PROVIDER_TYPE
id
code
label
last_update


Añadir datos de proveedor
<include>
Añadir proveedor
<extend>
Crear proveedor
<include>
Hacer pedido
<include>
Añadir datos de pedido
<include>
Añadir linea de pedido
Modificar activo
<include>
Añadir nuevos datos
Gestor de pedidos
Asignar datos contables
Cargar en MDM
<extend>
<extend>
Crear centro de coste
Crear cuenta contable

5.3.2 Sistema SCM

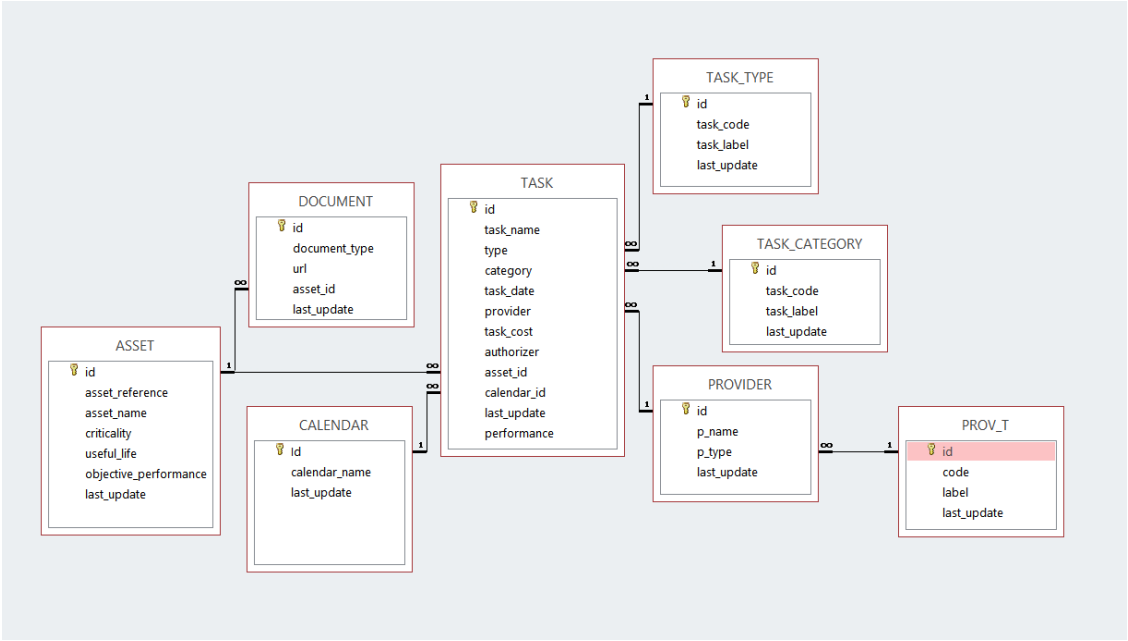
A continuación, se muestra la base de datos definida para el sistema SCM y los diagramas de casos de uso para la implementación de los formularios:

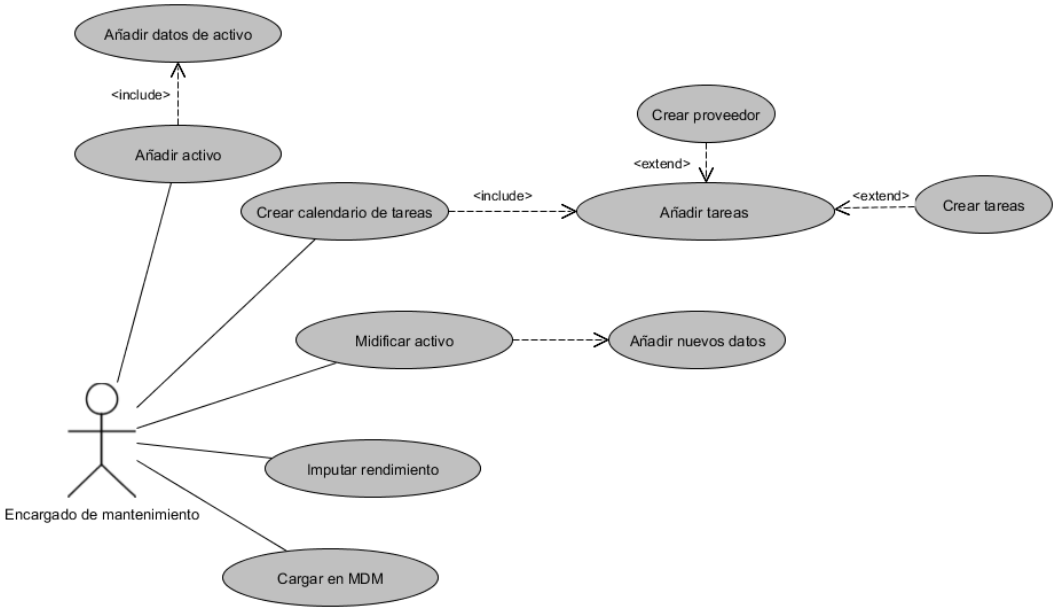




5.3.3 Sistema CMMS

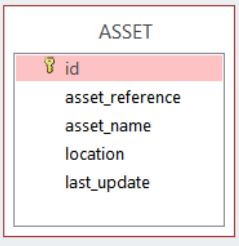
A continuación, se muestra la base de datos definida para el sistema CMMS y los diagramas de casos de uso para la implementación de los formularios:

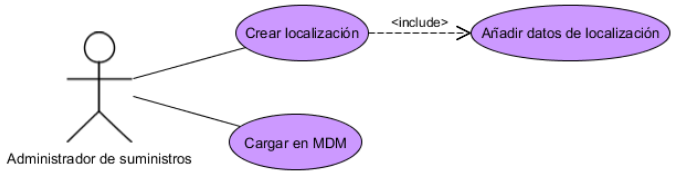




5.3.4 Sistema GIS

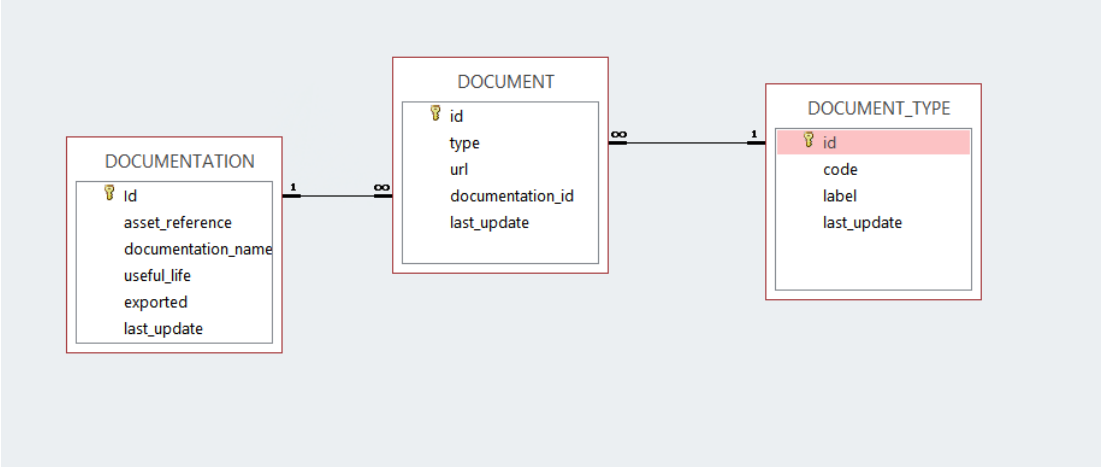
A continuación, se muestra la base de datos definida para el sistema GIS y los diagramas de casos de uso para la implementación de los formularios:

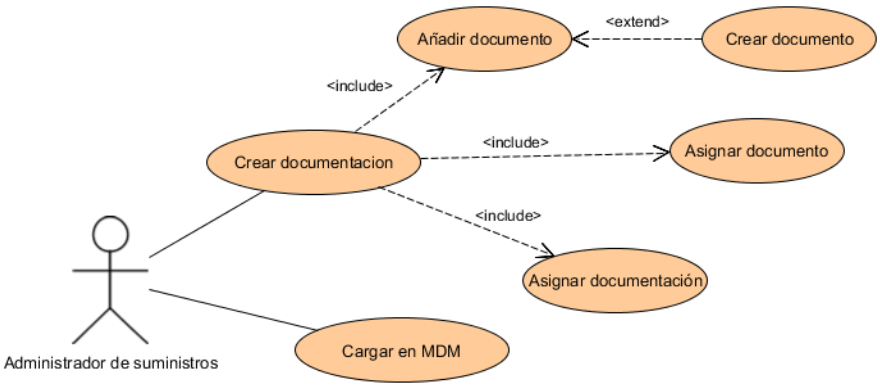




5.3.5 Sistema CMS

A continuación, se muestra la base de datos definida para el sistema CMS y los diagramas de casos de uso para la implementación de los formularios:





5.3.6 Método de carga

Como se puede observar, todos los sistemas aportan datos al MDM mediante un método que los carga. Esto se realizará en un formulario con un botón, el cual disparará el método que cargará todos los datos de su sistema en las tablas Landing de MDM.

Este método, implementado en Visual Basic, hará una conexión de tipo ADODB con los siguientes atributos:

- Provider= SQLOLEDB
- Data Source= 192.168.2.101, 11433
- Initial Catalog= ORS_TFG
- User Id= ORS_TFG

Para cada tabla, definiremos en SQL una sentencia INSERT para insertar en cada tabla Landing correspondiente del MDM los datos que correspondan, junto con el sistema y la tabla destino, que se imputarán de forma explícita.

5.4 Prueba unitaria

En este apartado se describirá la 'prueba unitaria', consistente en un cuadro de mando, confeccionada para la comprobación de correcto funcionamiento del sistema desarrollado. De esta forma, garantizaremos que el cliente podrá implantar su propio sistema BI.

El cuadro de mando implementado permite visualizar los Golden Records, consultando al MDM mediante los servicios ofrecidos por el servidor (el cual hemos configurado con anterioridad), y mostrar una visión unificada de la información. Esta sencilla herramienta únicamente interaccionará con el usuario en el momento de elegir el activo objeto de análisis.

5.4.1 Definición de indicadores

En este caso, los indicadores que, junto al cliente, se han decidido que sean:

- Vida útil esperada
- Fecha puesta en funcionamiento
- Coste de adquisición
- Rendimiento objetivo

- Criticidad
- Coste de mantenimiento preventivo
- Coste de mantenimiento imprevisto
- Coste de desmantelación

Estos indicadores se han elegido en base a una necesidad muy concreta, pero podrían cubrirse muchas más necesidades si el cliente lo requiriera simplemente seleccionando otros indicadores.

5.4.2 Interfaz gráfica

La interfaz consistirá en tres gráficas que mostrarán, de forma esquemática, los datos para su evaluación y análisis.

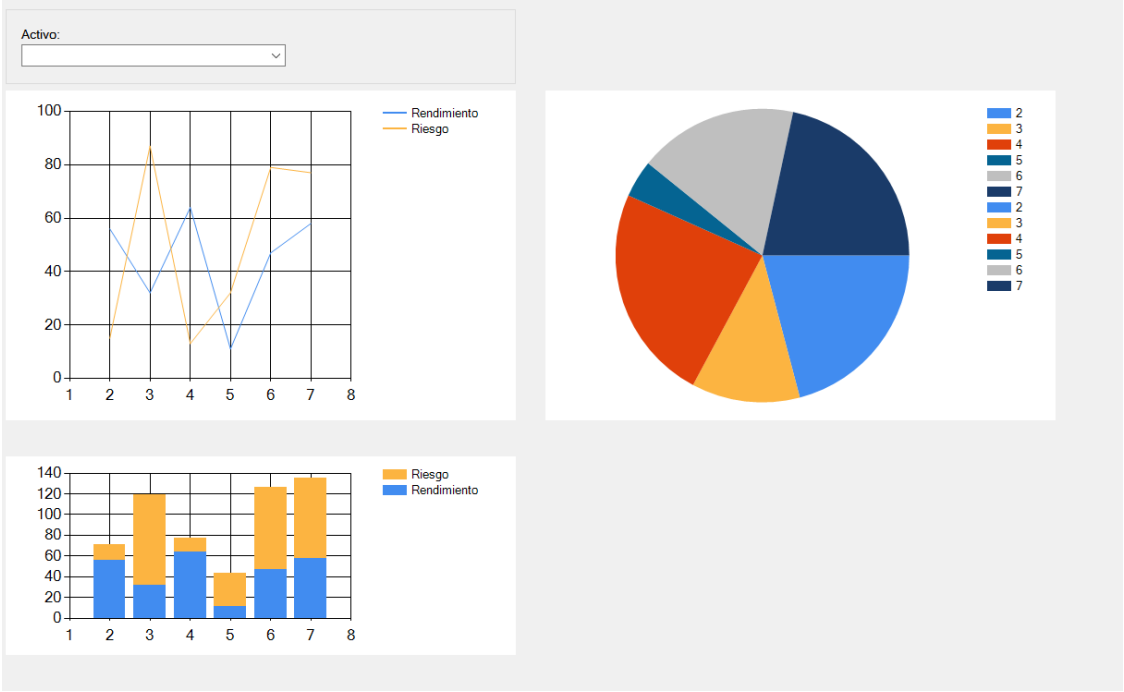


fig. 46 Interfaz del cuadro de mando

- *Gráfica lineal*: Vida útil. Se muestra la relación entre rendimiento monetario del activo y sus costes (de adquisición, mantenimiento, etc.). Permitirá identificar el momento en el que el activo ya ha sido amortizado y, por tanto, debe ser desmantelado y sustituido por otro.

- *Gráfica de barras*: Uso de activos. Se muestra la vida útil estimada para cada activo y el tiempo que llevan usándose. Esto permitirá llevar un control de tiempos que ayudará a planificaciones de cara a posibles sustituciones.
- *Gráfica de sectores*: Costes. En esta se visualiza qué tipo de costes y qué cantidades se invierten en cada uno, lo que proveerá de conocimiento acerca del dinero que está gastando la empresa en sus activos.

5.5 Ejemplos de uso de MDM

En el anexo B, se describirán ejemplos de uso reales que se podrían dar en este tipo de proyectos, según los siguientes tipos:

- *Change Request*: Cambios que pide el cliente sobre los requerimientos iniciales.
- *Error*: Situaciones en las que la herramienta detecta errores o advertencias.
- *Test*: Planteamiento de posibles escenarios y alternativas y sus repercusiones.

6. Conclusiones

Podemos afirmar que, después del trabajo realizado, hemos logrado crear un sistema capaz de integrar múltiples sistemas (fuentes de datos) que manejan distintas versiones de una parte de una organización, y poder extraer información consolidada y fiable que capacite la toma de decisiones en base a la mejor versión de la realidad de los datos de la empresa.

Tras un proceso de desarrollo largo, con una herramienta capaz de implantar una tecnología tan potente como la de MDM, y mirando a través de la ventana de cuadro de mando, hemos conseguido mostrar otras formas emergentes de tratamiento de datos.

Después del desarrollo y puesta en funcionamiento, se han alcanzado las siguientes metas:

- Simular un entorno de sistemas de información y su interacción, capacitando la reproducción de problemas, errores, incoherencias, inexactitudes, duplicidades, etc., derivado del manejo continuo y prolongado de información distribuida, resultado de transacciones que involucran a diversos sectores de la organización.
- Crear un sistema con tecnología MDM que se encargue de resolver el problema de la heterogeneidad de los datos de la organización, describiendo su implementación e implantación, así como mostrar los resultados que este proporciona.
- Desarrollar una herramienta capaz de ilustrar las bondades del sistema implantado e introducir posibles usos que podrían hacerse de los datos.

El hecho de haber desarrollado el presente trabajo en un entorno laboral y con herramientas que se utilizan en proyectos reales ha sido sumamente enriquecedor, en cuanto a formación y experiencia laboral se refiere, así como una gran oportunidad de poner en práctica gran parte de los conceptos y conocimientos adquiridos durante los años de estudio en la carrera.

6.1 Retos y problemas

Uno de los principales problemas que es frecuente encontrar es el relativo a la comunicación con el cliente. El desarrollo se empezó en base a unos requerimientos iniciales, proporcionados por el cliente ficticio diseñado por un compañero de la empresa. Como sucede comúnmente en entornos reales, el cliente cambió de opinión con respecto al planteamiento que se hizo al inicio, lo que derivó en el cambio de

especificaciones y herramientas empleadas. Además, hubo que realizar un rediseño de gran parte del sistema que ya estaba en fase de desarrollo. Estas circunstancias a menudo derivan en el fracaso del proyecto, pero gracias al uso de metodologías ágiles fue fácilmente resuelto.

También hubo que resolver cuestiones tales como la definición de datos críticos y su tratamiento, proceso que, de nuevo, no podría haberse realizado sin una comunicación activa y continua con el cliente.

La metodología de aprendizaje fue, a su vez, un reto importante. Usar una tecnología de vanguardia tiene implicaciones tales como la falta de información (para el desarrollo, solución de problemas, etc.), así como la dificultad en encontrar experiencias similares.

Otro problema encontrado fueron las actualizaciones y puestas al día, un proceso habitual y necesario en la mayoría de organizaciones y sistemas. Como ya se ha descrito en puntos anteriores, la actualización del sistema operativo en la estación de trabajo derivó en una inutilización parcial del sistema que se estaba desarrollando con el servidor, lo que se resolvió con cambios en el adaptador de red.

Pero, el principal reto y el más gratificante, fue el poder empezar a desenvolvernos en ambientes de trabajo reales, aprendiendo de experiencias de profesionales con años de práctica y enfocando todo lo aprendido hacia nuestra futura vida laboral.

6.2 Líneas de trabajo abiertas

En base al trabajo realizado, hemos llegado a la conclusión que puede ser la base a otros trabajos orientados más en aumentar la complejidad de las fuentes y estudiar su consolidación o, incluso, aumentar la complejidad de la información y el conocimiento que se extraiga de los datos. Algunas líneas de trabajo que podrían seguirse son:

- Estudiar casos de implantación de MDM en empresas reales y su repercusión en el rendimiento de la misma, así como de su posición en el mercado antes y después de la implantación de dicho sistema.
- La implantación en un entorno simulado del mismo sistema con datos reales, ya sea con el modelo de datos actual o con uno adaptado, lo que permitiría llegar a comprender mejor cómo actúa el sistema y la metodología empleada.
- Complementar el sistema con un proceso ETL (Extract, Transform and Load) previo, para mostrar cómo interactuarían otras tecnologías con este sistema que, en ciertos entornos de trabajo, se hace inevitable.

- Completar del cuadro de mando aumentando el número de indicadores seleccionados y el número de gráficas con la información de la que ya se dispondría en el sistema.
- Aumentar la funcionalidad del cuadro de mando con la capacidad de generar informes.
- Extender el sistema hacia un cuadro de mando integral (CMI).
- Estudiar la escalabilidad del sistema mediante la adhesión escalonada de nuevos sistemas, campos o tablas.
- Montar un sistema BI profesional que se alimente de los Golden Records del MDM.

Anexo A: Bibliografía

Amado Salgueiro (2001). Indicadores de Gestión y Cuadro de Mando. Editorial Diaz de Santos.

www.informatica.com (27/07/2015). Business Intelligence, informes y análisis. http://international.informatica.com/es/solutions/enterprise-data-integration-and-management/master-data-management/business-intelligence-reporting-and-analytics/

Informatica (2010). MDM y calidad de datos para el data warehouse. http://international.informatica.com/es/Images/7209-enable-accurate_eb_ES.pdf

www.informatica.com (23/08/2015). The Forrester Wave: Soluciones de gestión de datos maestros. http://international.informatica.com/la/multi-platform-mdm-forrester-wave/

Espiñera, Sheldon y Asociados (2008). Boletí de Asesoría Gerencial: La Inteligencia de Negocios. https://www.pwc.com/ve/es/asesoria-gerencial/boletin/assets/boletin-advisory-edicion-10-2008.pdf

www.ibm.com (12/08/2015). Posicionamiento MDM. http://www.ibm.com/developerworks/ssa/local/data/posicionamento_mdm/

www.ibm.com (12/08/2015). Desempeño de activos. https://www.ibm.com/developerworks/community/blogs/b35561d9-e0ef-48e0-b455-001f4a64b4da/entry/la_importancia_del_desempe_c3_b10_de_los_activos9?lang=en

msdn.microsoft.com (13/08/2015). C# y .NET Framework. https://msdn.microsoft.com/es-es/library/z1zx9t92.aspx

www.lantares.com (28/07/2015). MDM. http://www.lantares.com/blog/master-data-management

es.wikipedia.org (27/07/2015). Inteligencia empresarial. https://es.wikipedia.org/wiki/Inteligencia_empresarial

www.sinnexus.com (27/07/2015). Business Intelligence. http://www.sinnexus.com/business_intelligence/

www.pgconocimiento.com (27/07/2015). Business intelligence. http://www.pgconocimiento.com/Soluciones/que-es-bi.html

www.businessintelligence.info (29/07/2015). Necesidad de un Sistema BI. http://www.businessintelligence.info/docs/varios/raul-benet-sistema-bi-necesario-en-las-empresas.pdf

www.informationbuilders.es (27/07/2015). Master Data Management. http://www.informationbuilders.es/mdm-master-data-management

Anexo B: Ejemplos de uso de MDM

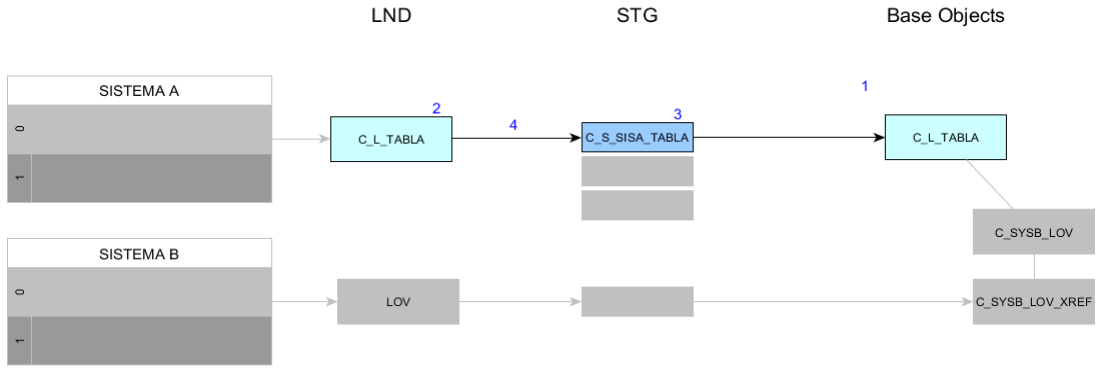
B.1 Mapeo de nuevos campos

B.1.1 Tipo: Change Request

Este caso de uso se centra en la adición de dos nuevos atributos en las tablas Landing para mapearlos al objeto que le corresponde, uno de ellos siendo un valor de una lista de valores que provee otro sistema. El cliente deberá proporcionar los nuevos atributos, así como la fuente que los provee. Asumiremos que los atributos ya existen en la tabla correspondiente en los Base Objects.

B.1.2 Análisis

Los campos que se añaden son 'campo1' y 'campo2' en la tabla 'TABLA' y son poblados por el sistema 'SISTEMA A'. El siguiente diagrama muestra el flujo de interacciones que hay que habría que realizar con el sistema:

DIAGRAMA


PASO	DESCRIPCIÓN
1	Se busca en la tabla destino los campos que habría que añadir a la Landing. Se apunta su tipo y examinan las referencias a otras tablas que pudiera tener. En este caso, 'campo2' apunta a una tabla de lista de valores (LOV) llamada SYS_B_LOV.
2	Se añaden los atributos en la Landing (o Landings si hubiera más de una).
3	Se añaden los atributos a la Stagging. Como 'atributo2' hace referencia a una LOV, tendremos que indicar 'SISTEMA B' en el campo Lookup System, 'SYS_B_LOV Cross_reference' en el campo Lookup Tbale y Pkey Source Object en el campo Lookup Column.

4	En el Mapping, se une 'atributo1' y 'atrubito2' de la Landing a la Stagging directamente, puesto que se asume que transformación se ha llevado a cabo previamente mediante ETL.

B.2 Listas de valores suministradas por fuentes

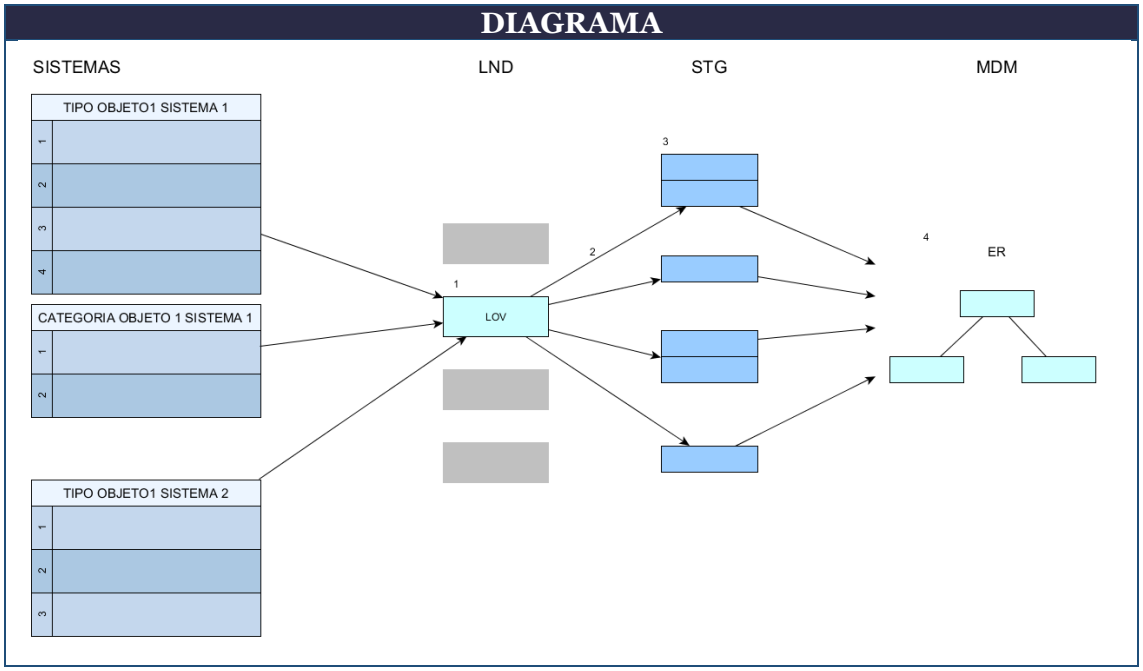
B.2.1 Tipo: Test

Este caso de uso trata la carga de listas de valores en MDM cuando las tablas Landing se cargan desde distintos sistemas que podrá proveer o no los mismos valores.

Cada fuente dispone de sus propias tablas de Listas de Valores (Lists Of Values*, de ahora en adelante LOV).

Inicialmente se asumía que las listas de valores los proveía el propio MDM desde una tabla con datos insertados a mano, pero cada fuente tiene su propia versión de estos datos, así que se opta por que sean las propias fuentes los que provean estos datos.

B.2.2 Análisis



PASO	DESCRIPCIÓN
1	Cada fuente inserta los datos que se consideren como listas de valores a la tabla de Lista de Valores de las Landing en MDM.
2	El sistema deberá identificar qué sistema provee cada dato y cuál es su tabla destino a fin de insertarlo en la Staging correcta.
3	Cada dato es insertado en la Stagging a la que pertenece.
4	Se insertan los datos en las tablas de Base Objects a las que deberían pertenecer.

B.2.3 IMPACTO

Únicamente se dispone de una tabla de LOV cuyos datos se deberán repartir entre varias tablas dentro de los Base Objects. Esto, junto al hecho de que puede haber un valor suministrado por más de un sistema, hará que se vean afectadas tanto las tablas Stagging como los mappings.

B.3 Incremento de la precisión de un atributo

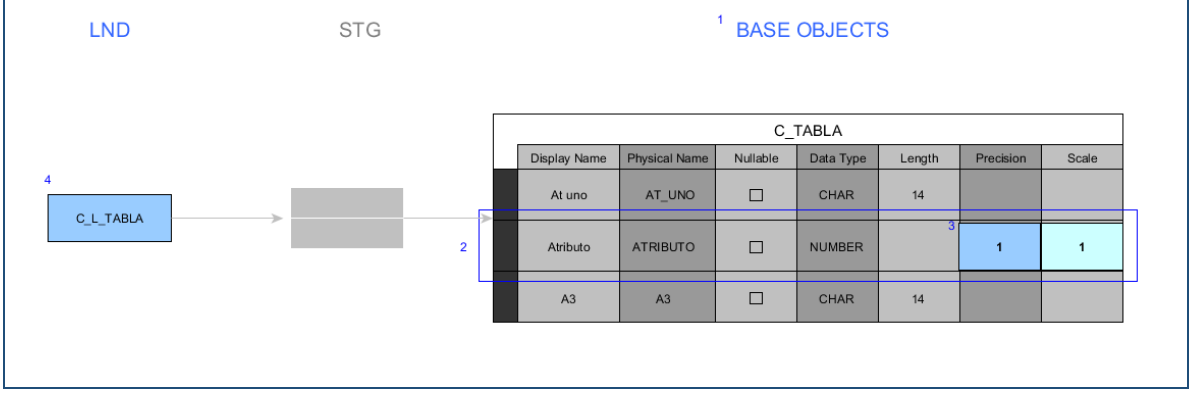
B.3.1 Tipo: Change request

Este caso de uso viene dado por un cambio de los requerimientos que da el cliente. Se trata de subir la precisión de ciertas columnas de ciertos base objects.

B.3.2 Impacto

El cliente ha pedido que el atributo 'ATRIBUTO' del objeto 'TABLA' pase de NUMBER (1,1) A NUMBER (3,2)*. Esto implica ir a los BASE OBJECTS, localizar TABLA y en ATRIBUTO cambiar el campo 'Precision' de 1 a 3 y el campo 'Scale' de 1 a 2.

DIAGRAMA



PASO	DESCRIPCIÓN
1	Se hace un *export*** antes de cualquier cambio, lo que permitirá volver hacia atrás en caso de cometer algún error.
2	Se busca el objeto, en este caso TABLA, en los BASE OBJECTS.
3	Se identifica el atributo NUMBER.
4	Se cambian los campos 'Precision' y 'Scale'.
5	Se repiten los pasos 1, 2 y 3 pero esta vez en la Landing o Landings correspondientes (es poco frecuente que haya más de una).
6	Se valida el ORS.

B.3.3 Impacto

Generalmente, al hacer un cambio en los BASE OBJECTS el sistema se encarga de aplicar el cambio en cascada, es decir, actualiza las Stagings de forma automática, pero cabe la posibilidad de que no suceda así. Aparte, cabe destacar todas las otras tablas que también podrían ser afectadas, muchas de ellas útiles a la hora de rastrear problemas:

TABLA AFECTADA	DESCRICIÓN
C_TABLA	Tabla del objeto en los Base Objects.
| |-> C_TABLA_CTL	
| |-> C_TABLA_EMI	
| |-> C_TABLA_EMO	
| |-> C_TABLA_HCTL	
| |-> C_TABLA_HIST	
| |-> C_TABLA_HMRG	
| |-> C_TABLA_HVXR	

| |-> C_TABLA_MTCH	
| |-> C_TABLA_STRP	
| |-> C_TABLA_VCT	
| |-> C_TABLA_VXR	
| |-> C_TABLA_XREF	Tabla de referencias cruzadas que contiene los datos que proveen los distintos sistemas.
| |-> C_TABLA_HXREF	Tabla del histórico de XREF
| |-> C_S_SIS1_TABLA	Stagging de TABLA para el sistema Sistema1.
| |-> C_S_SIS1_TABLA_OPL	
| |-> C_S_SIS1_TABLA_PRL	Tabla Landing previa que se crea al definir el Mapping desde la Landing a la Stagging.
| |-> C_S_SIS1_TABLA_RAW	Histórico que se crea cuando la propiedad Audit Trail está ativada.
| |-> C_S_SIS1_TABLA_REJ	Contiene los registros que han sido rechazados.

En este caso, el ORS no se ha validado por este problema y se deben cambiar a mano tanto las Staggins como los Mappings.

B.3.4 Solución

Los pasos a seguir para atajar el problema son los siguientes:

PASO	DESCRIPCIÓN
1	Acudir a los Mappings y eliminar el link del atributo en cuestión y apuntar el trust.
2	Se abren las Staggings, se apuntan los campos que estén o no seleccionados, se deselecciona el atributo y se guarda. Seguidamente, se vuelve a incluir, se vuelven a selecciona los demás campos a como estaban y se vuelve a guardar.
3	Se rehace el link en los Mappings.
4	Se reconfigura el trust a como estaba antes de los cambios.

B.4 El resultado del mapeo de PKEY_SRC_OBJECT es nulo

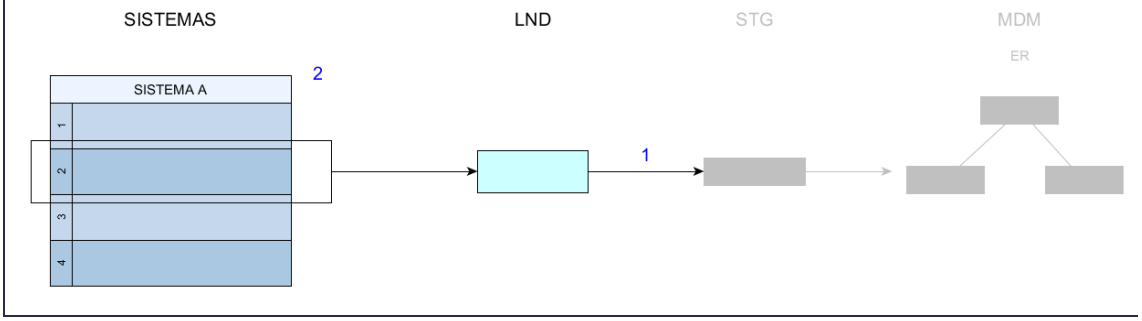
B.4.1 Tipo: Error

El problema radica en que, durante el proceso de carga, detecta que se está proporcionando un valor nulo desde la tabla Landing.

B.4.2 Análisis

Durante el proceso de carga, se genera el error cuando se detecta que la tabla Stagging 'C_S_SISA_TABLA' tiene su campo pkey_src_object con un valor nulo. En el siguiente diagrama se puede ver los pasos que hay que seguir para solucionar el problema:

DIAGRAMA



PASO	DESCRIPCIÓN
1	Inspeccionar el Mapping que corresponda para saber qué campo de de la tabla Landing tendría que proporcionar el dato.
2	Avisar al responsable de la carga de datos al MDM para hacerle saber que no se está insertando ningún dato en el campo que tendría que proporcionar el valor de pkey_src_object.

B.5 Fecha nula o perteneciente al futuro

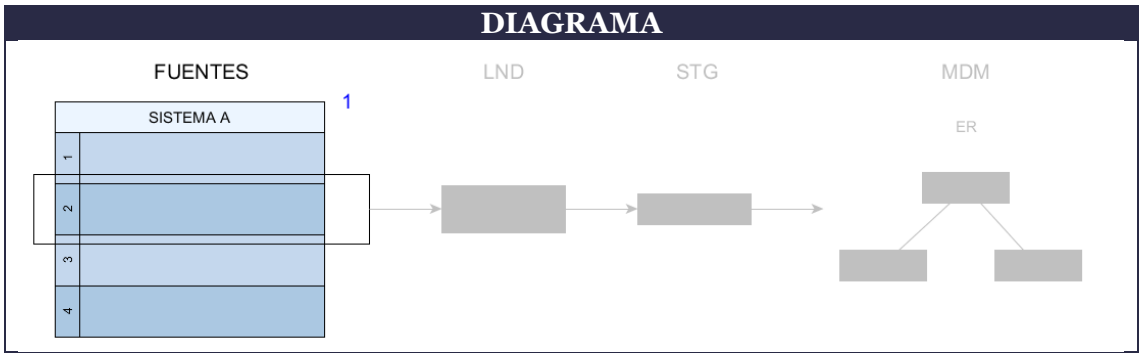
B.5.1 Tipo: Error

Este caso de uso aparece cuando el campo LAST_UPDATE_DATE tiene un valor nulo o su fecha no es ni presente ni pasada, sino perteneciente al futuro, por lo que se considera erróneo.

Puede aparecer en un registro descartado al lanzar un Batch Group con la siguiente descripción: Date is either in the future ori s null. Junto con la descripción, podremos ver los campos del registro rechazado, entre ellos el campo LAST_UPDATE_DATE

B.5.2 Análisis

Ante este problema, que es principalmente derivado del proceso previo a la carga de datos, el primer y único paso (véase el diagrama que se muestra en la parte inferior) es avisar al personal encargado de este proceso previo para que solucionen el problema, es decir, que el sistema en cuestión cargue en la tabla Landing correspondiente el registro con el campo LAST_UPDATE_DATE con un valor válido.



B.6 Clave ajena con valor nulo

B.6.1 Tipo: Error

En este caso nos encontramos con que uno de los campos, que es una clave ajena a otra tabla, es nulo.

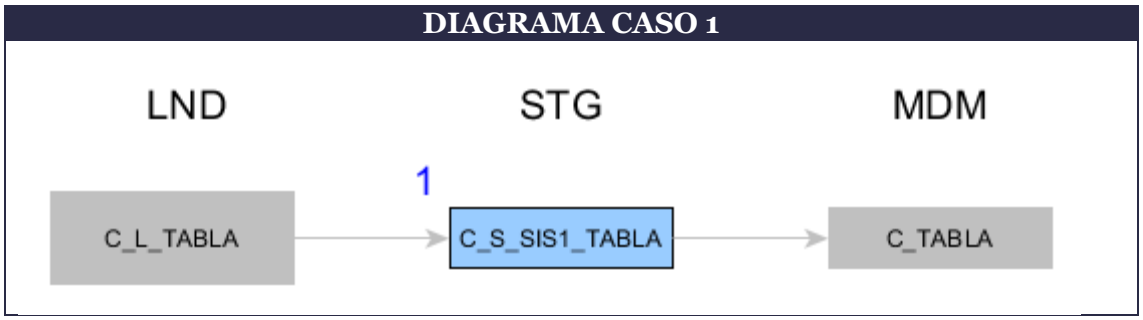
Al ejecutar el Batch Group, aparecerá un registro descartado con el código de error siguiente: Foreign key [CATEGORY] cannot have a null value.

B.6.2 Análisis

El error aparece al intentar cargar la tabla 'TABLA' de los Base Objects un registro con un campo 'CLAVE AJENA' que hace referencia a otra tabla y que tiene valor nulo. En este caso puede haber dos causas: que sea un error en la configuración del Hub del MDM o que venga dado por un error del sistema que provee ese dato.

B.6.1.1 Caso 1: Error de configuración en el Hub de MDM

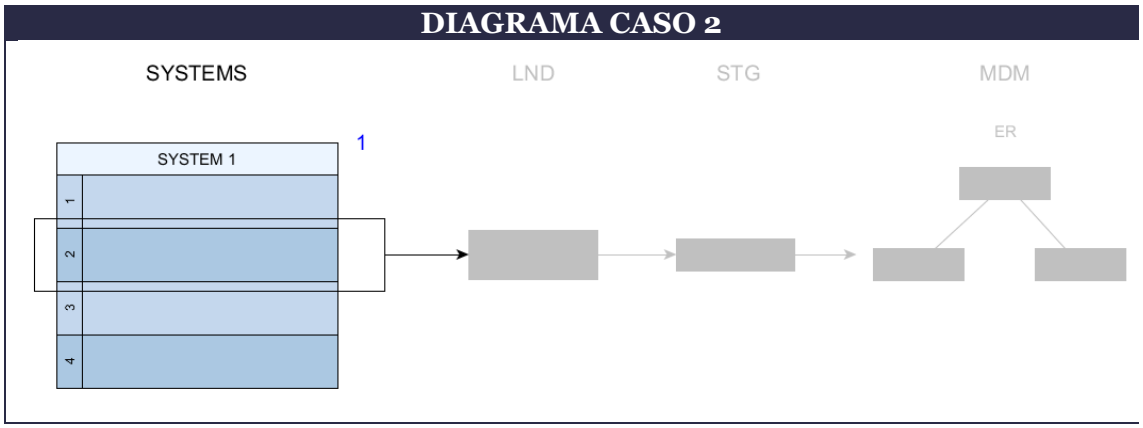
En este caso, se asume que los datos que vienen del sistema 'SISTEMA 1' son correctos y que no aporta ningún dato para el campo 'CLAVE AJENA' porque es un campo que no existe en su sistema.



El primer y único paso (1) que hay que realizar es permitir la inserción de nulos para ese campo dado que, aunque la herramienta se comporte como una base de datos, no lo es. Basta con ir a la definición de la tabla Stagging 'C_S_SIS1_TABLA' para cambiar su configuración (pestaña Columns). Hay dos opciones, o descartar el campo 'CLAVE_AJENA' o señalar Allow Null Foreign Key. Si el en la fuente no existe el campo, se descarta directamente en la tabla Stagging.

B.6.1.2 Caso 2: Error del sistema que provee el dato

Este apartado sirve de solución cuando el problema viene de la fuente que provee el dato.



El paso que hay que realizar (1) es contactar con el encargado de hacer la carga de datos previa desde la fuente al MDM.

B.6.3 Impacto y solución

En el caso 1 expuesto en el apartado anterior, aparte de la tabla Stagging que queremos modificar, se verá implicado el Mapping, ya que se quiere descartar un campo que está unido entre una tabla Landing y la tabla Stagging. Lo que habrá que hacer es eliminar el link para, posteriormente, descartar el campo en la Stagging.

B.7 Registros duplicados en las tablas Landing

B.7.1 Tipo: Error

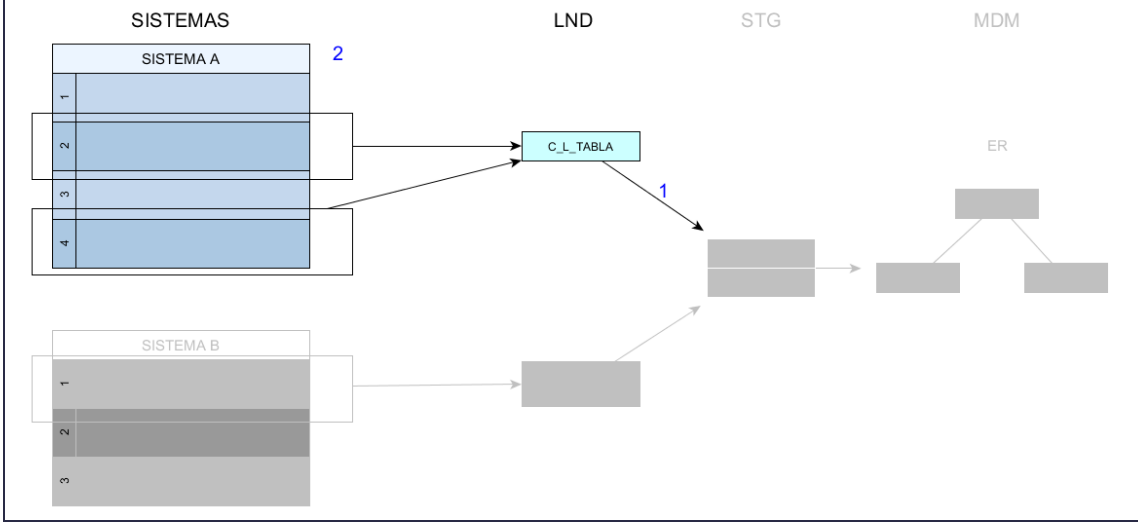
Es un problema fruto de una carga errónea de datos en las tablas Landing, ya que el sistema encuentra dos filas iguales.

La forma en la que se presenta este problema es al ejecutar los Batch Groups. Aparecerá como la causa de alguna grabación rechazada (rejected record).

B.7.2 Análisis

Este problema surge cuando el sistema 'SISTEMA A' inserta dos veces el mismo registro en la landing 'C_L_TABLA'. Esto ocasiona que el sistema no pueda mapear el registro a la tabla Staging que corresponda. La solución viene dada en dos pasos:

DIAGRAMA



PASO	DESCRIPCIÓN
1	En el Mapping M_SISA_TABLA, en la pestaña Query Parameters, se señala la opción Enable Distinct. Esta solución permite seguir trabajando ya que se consigue mapear como un único regitro.
2	Dado que no se conoce la causa por la que hay duplicados en las Landing (puede ser algo normal o un debido a algún error previo a la carga de datos), se le deberá comunicar al personal encargado del proceso de carga de datos en las Landing este problema, para que tengan constancia del fallo y facilitarles el que puedan solucionar algún posible error.

B.8 No se encuentra el valor de la Lookup

B.8.1 Tipo: Error

En este caso nos encontramos con que el campo de lookup no se puede encontrar, es decir, hay un campo que es una clave ajena en la que no se está encontrando su valor en el campo de la tabla especificada como lookup.

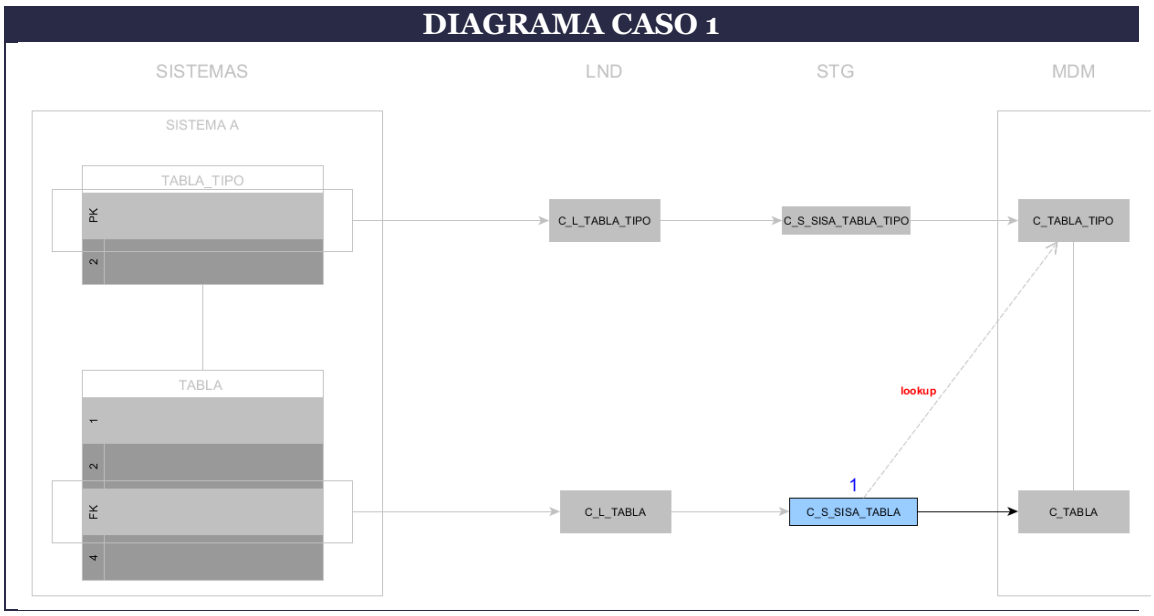
En el registro descartado al hacer la carga nos encontraremos con el siguiente mensaje: Cannot find lookup value 'valor' form column CAMPO.

B.8.2 Análisis

Este error puede ser causado, principalmente, por dos causas: cuando el campo es una clave ajena en la fuente que hace referencia a otra tabla de esa fuente o de otra (que se señalará como la lookup) o cuando el dato de ese campo no es una clave ajena en la fuente, por lo que no está apuntando a ninguna tabla.

B.8.2.1 Caso 1: El dato del campo es una clave ajena

La tabla 'TABLA' tiene un campo 'tipo_campo' que es una clave ajena a la tabla 'TABLA_TIPO' del mismo sistema ('SISTEMA A').



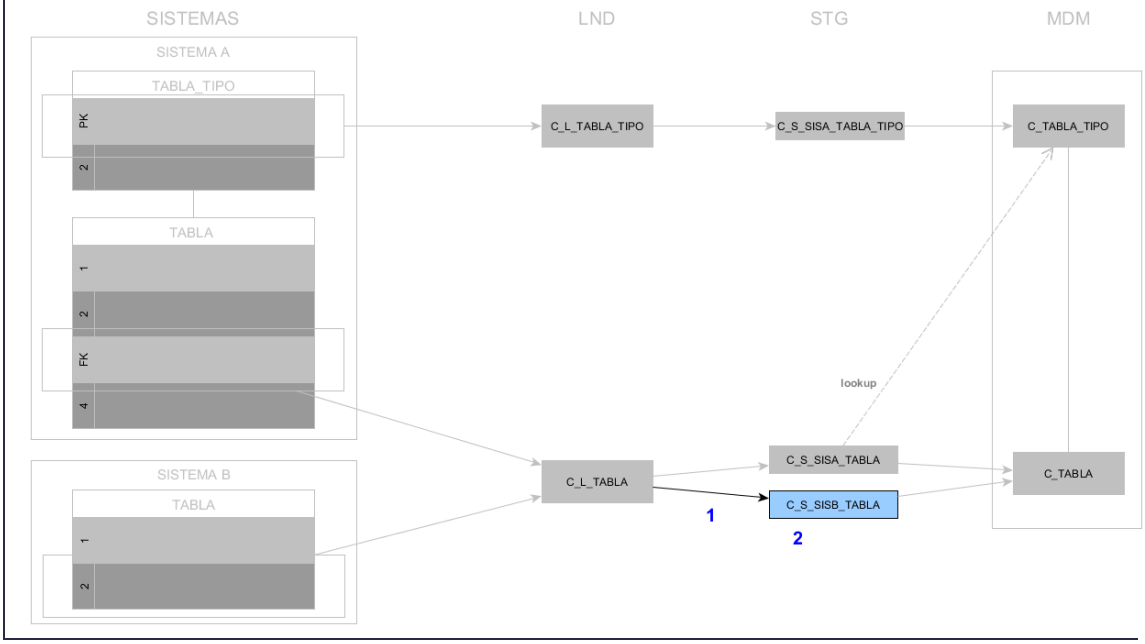
El paso (1) para resolver este error es ir a la definición de la tabla Stagging 'C_S_SISA_TABLA' y modificar el lookup value con el sistema y el campo correctos o incluirlo si no existiera. Como en todo momento hemos asumido que el dato de la clave ajena no es suministrado por el MDM sino por un sistema externo (el mismo sistema en este caso), en la definición de la tabla Stagging 'C_S_SISA_TABLA' (pestaña Columns) editaremos para el campo 'tipo_campo' su lookup señalando como Sistema Lookup

'SISTEMA A', como Tabla Lookup 'TABLA_TIPO Cross-reference' y como Columna Lookup 'Pkey Source Object'.

B.8.2.2 Caso 2: El dato del campo es un string

La tabla 'TABLA' en el sistema 'Sistema B' tiene un campo 'tipo_campo' que es un nombre que no hace referencia a otra tabla. En este caso, el MDM no puede encontrar esa referencia porque no es una referencia de ninguna tabla que se haya añadido con anterioridad.

DIAGRAMA CASO 2



Puesto que 'SISTEMA A' provee el dato maestro, se procederá a impedir que 'SISTEMA B' inserte el campo 'tipo_campo'. Estos son los pasos que habrá que hacer para solucionar el problema:

PASO	DESCRIPCIÓN
1	Quitar en el Mapping el link del campo 'tipo_campo'.
2	Descartar en la tabla Stagging 'C_S_SISB_TABLA' el campo 'tipo_campo'

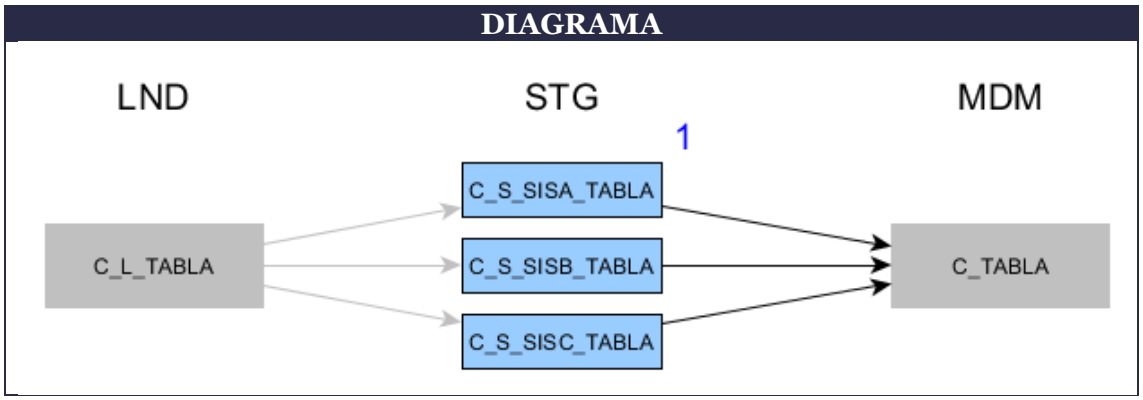
B.9 Actualización por LUD

B.9.1 Tipo: Test

La actualización por Last Update Date (de ahora en adelante LUD) hace referencia a la actualización de campos (que no registros) por la última fecha en la que se actualizó un registro en la fuente.

B.9.2 Análisis

El sistema 'Sistema A' actualiza el campo 'campo' de la tabla 'TABLA'. Como es el cambio más reciente, en este caso queremos que sea el que prevalezca.



Se deberá cambiar la configuración de todas las tablas de Stagging (en 'C_S_SISA_TABLA', 'C_S_SISB_TABLA' y 'C_S_SISC_TABLA'). En la pestaña 'Settings' seleccionaremos 'Enable delta detection' y 'Detects deltas through a date column' en el que seleccionaremos el campo Last Update Date.

Al actualizar un campo de un registro en la fuente y cargarlo al MDM, después de ejecutar el Batch Group deberemos ver que se han actualizado los campos según la versión del campo más reciente. Se asume que en cada sistema guarda la fecha de la última actualización de cada registro.

B.9.3 Impacto

El histórico del "delta detection" se guarda en la base de datos. Esto tiene implicaciones a la hora de hacer la carga inicial si ya se había hecho con anterioridad, ya que no sólo habrá que eliminar los datos de las tablas de Landing, Stagging, Base Objects y XREF.

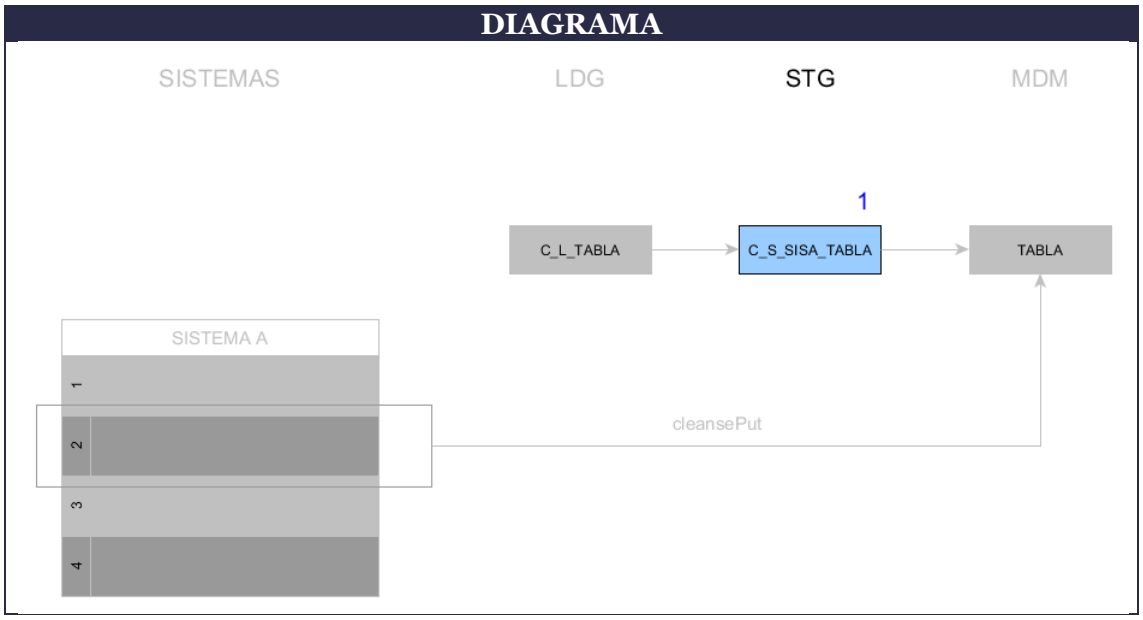
B.10 Allow Null Update

B.10.1 Tipo: Test

Este caso de uso se centrará en probar la propiedad 'Allow Null Update' de las tablas Staging, el cual deberá permitir que un campo que tiene valor podrá actualizarse a nulo, cosa que de normal no se podría.

B.10.2 Análisis

El test se realiza sobre la tabla 'TABLA', donde desde el sistema 'SISTEMA A' actualizará el campo 'campo'.



El único cambio que se realizará (1) será en la definición de la tabla Staging 'C_S_SISA_TABLA'. En el apartado Columns, se elige el campo (en este caso 'campo') con el que se quiere hacer la prueba y se marca la casilla 'Allow Null Update'. Entonces, se actualiza un registro desde 'SISTEMA A' con soapUI utilizando el método 'cleansePut' en el que su campo 'campo' está a nulo. Seguidamente, se comprueba que el registro se ha actualizado y que el campo 'campo' ha pasado a nulo.

B.11 Apply Null Values

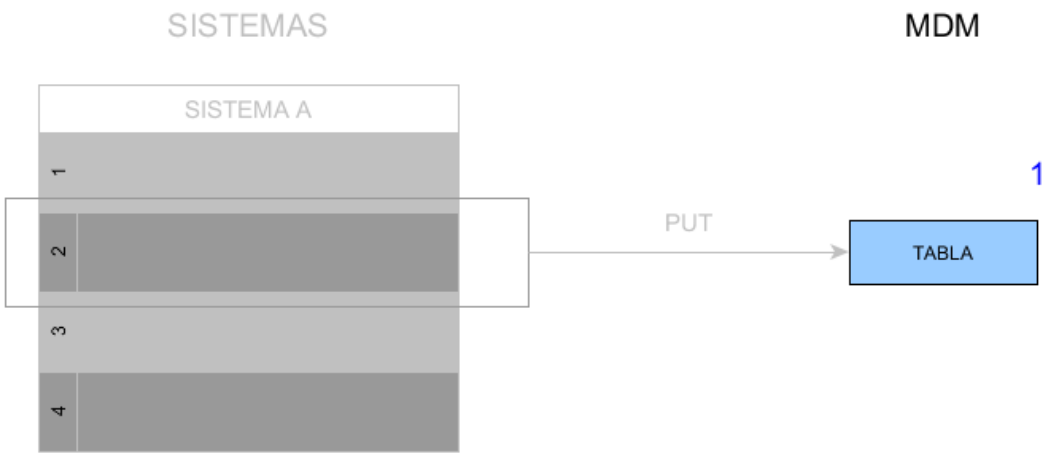
B.11.1 Tipo: Test

Este caso de uso se centrará en probar la propiedad 'Apply Null Values' de los Base Objects. Dicha propiedad permite actualizar un campo a nulo de una tabla directamente sobre una tabla de los Base Objects. Si esta propiedad se encuentra desactivada, prevalecerá siempre la versión del campo con valor no nulo, siempre que haya más de una versión de ese registro en las tablas de Cross Reference (XREF).

B.11.2 Análisis

El test se realiza sobre la tabla 'TABLA', donde desde el sistema 'SISTEMA A' se proveerá de un dato nulo para el campo 'campo'.

DIAGRAMA



El único cambio que se realizará (1) será en la definición de 'TABLA' en los Base Objects. En el apartado Columns, se elige el campo con el que se quiere hacer la prueba y se marca la casilla 'Apply Null Values'. Entonces, se inserta un dato desde 'SISTEMA A' con soapUI utilizando el método 'put' y se comprueba que el dato nulo se ha insertado correctamente. Hay que recordar que en C_TABLA_XREF deberá de haber más de una versión del mismo registro, ya que si sólo existe una y con valor nulo, 'campo' se actualizará a nulo. La siguiente tabla muestra el valor con el que se actualizaría el objeto dependiendo de los registros y su valor en 'campo' en las tablas XREF cuando la casilla 'Apply Null Values' está desactivada.

REGISTRO 1	REGISTRO 2	VALOR ACTUALIZADO
NULL	-	NULL
Valor 1	-	Valor 1
NULL	Valor 2	Valor 2
NULL	NULL	NULL
Valor 1	Valor 2	LUD

En la siguiente tabla se muestra el valor con el que se actualizaría el objeto dependiendo de los registros y su valor en 'campo' en las tablas XREF cuando la casilla 'Apply Null Values' está activada.

REGISTRO 1	REGISTRO 2	VALOR ACTUALIZADO
NULL	-	NULL
Valor 1	-	Valor 1
NULL	Valor2	LUD
NULL	NULL	NULL
Valor 1	Valor2	LUD

B.12 Trust

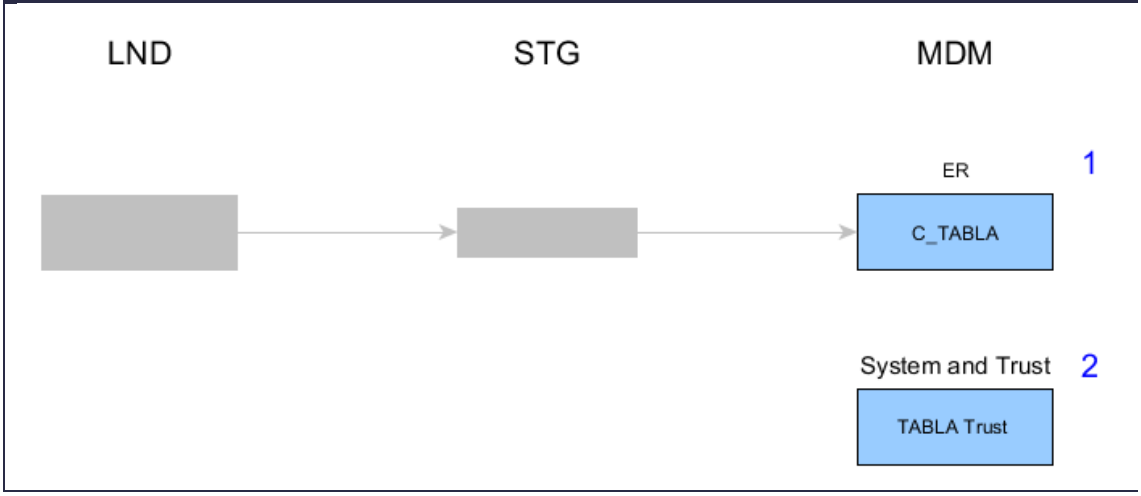
B.12.1 Tipo: Test

La propiedad 'trust' sirve para poner una prioridad de cara a la elección que hace el MDM del campo que prevalece sobre el resto de versiones del registro proporcionadas por los distintos sistemas. El trust se deberá tener en cuenta junto con otros parámetros como el LUD y la actualización de campos a nulo.

B.12.2 Análisis

Se quiere configurar para el campo 'columna' de la tabla 'TABLA', que es proporcionado por sistemas 'SISTEMA A', 'SISTEMA B' Y 'SISTEMA C', su 'trust' para cada uno de estos sistemas.

DIAGRAMA



Para ello, estos son los pasos que hay que realizar:

PASO	DESCRIPCIÓN
1	En la configuración de 'TABLA' en los Base Objects, en el apartado Columns, se señala para el campo 'columna' la propiedad 'Trust'.
2	En el apartado System and Trust, en el apartado Trust, se despliega 'TABLA' y se accede a las propiedades de 'columna'. Aparecerán, para cada sistema que provee el dato, una serie de propiedades: Maximum (el valor de trust máximo), Minimum (el valor de trust mínimo), Units (unidad de tiempo), Decay (la duración del trust después de la cual desaparece) y Graph Type (el tipo de gráfica).

La configuración dependerá de las necesidades del cliente. Inicialmente, todas deberían tener el mismo trust y el mismo valor máximo y mínimo. Aunque si, por ejemplo, hay un sistema que provee el dato sólo en la carga inicial, éste podría tener un trust máximo y mínimo más alto. Cuando el trust mínimo es menor que el máximo, significa de disminuirá con el tiempo, y puede suceder que otro sistema que tuviera inicialmente un trust inferior para el mismo campo, acabe por tener mayor preferencia.

B.12.3 Impacto

También hay que tener en cuenta que los resultados dependerán de si la propiedad 'Apply Null Vales' se ha incluido. Se contemplan 4 posibles casos dependiendo de si se añade dicha propiedad o no. En la siguiente tabla se muestra cual será la preferencia de inserción en función de la configuración y el tipo de datos:

TRUST	APPLY NULL VALUES	PREFERENCIA DE INSERCIÓN
-	-	LUD, valores no nulos, valores nulos
-	X	LUD
X	-	Trust, LUD, valores no nulos, valores nulos
X	X	Trust, LUD