

Instituto de Computación – Facultad de Ingeniería
Universidad de la República

Visión global del proceso de construcción de datawarehouse

Nathalie Deppen
Paola Ricca
Diego Trías

Proyecto de Grado
Agosto, 2010

Tutores: Fernando Carpani, Regina Motz

Resumen

Desde hace años los sistemas de información se han tornado indispensables para las organizaciones, por esto, muchas organizaciones poseen y procesan constantemente grandes volúmenes de datos. En este escenario, el siguiente paso es explotar esos datos, pero para hacerlo, es necesario comprenderlos y transformarlos en información. Debido a la importancia y necesidad de acceso a dicha información, las organizaciones han emprendido grandes proyectos de datawarehouse con variado nivel de éxito, debido a que son proyectos conocidos por ser complejos y especialmente propensos a complicaciones. Parte de la complejidad de un proyecto en el área de datawarehouse radica en que se necesita gran conocimiento del negocio para transformar datos en información y en la dependencia de los datos y de los sistemas operacionales que mantienen los mismos. Muchas de las complicaciones encontradas al desarrollar un DW se dan al diseñar la carga de datos, tarea que suele hacerse al final del proyecto.

El presente proyecto presenta una guía que consta de cinco fases o pasos para construir un datawarehouse abarcando el relevamiento de requerimientos, la representación de la realidad de la empresa, la generación del diseño conceptual, el diseño lógico y la carga de datos necesaria para el correcto funcionamiento del sistema.

Palabras clave: Datawarehouse, Carga, Ontología, Proceso, Desarrollo, Trazabilidad

CONTENIDO

RESUMEN.....	3
INTRODUCCIÓN.....	13
.....	13
Motivación.....	13
Objetivos.....	13
Caso de estudio.....	14
Organización del documento.....	14
RESEÑA DEL PROCESO DE CONSTRUCCIÓN DEL DW.....	15
Concepto de DataWarehouse	15
Modelos multidimensionales.....	17
Rol de la metadata en el proceso de DW.....	20
CONTEXTO DE TRABAJO.....	22
MD4DW.....	22
Descripción de las Fases.....	23
Fase 1: Comprensión de la realidad.....	23
Fase 2: Modelado de Dominio.....	23
Fase 3: Modelado Conceptual Multidimensional.....	23
Fortalezas y debilidades.....	24
OWL2DSS.....	24
Fortalezas y debilidades.....	24
CMDM.....	25
Estructuras en CMDM.....	25
Fortalezas y debilidades.....	27
Peralta.....	28
Fortalezas y debilidades.....	28
Larrañaga.....	28
Fortalezas y debilidades.....	29
Bernstein.....	29
Fortalezas y debilidades.....	30
Visión general.....	31
CONCEPCIÓN GENERAL DE LA SOLUCIÓN PROPUESTA.....	32

Estudios preliminares para definir el camino a seguir en este trabajo.....	32
Propuesta 1:.....	32
Propuesta 2:.....	32
Mejoras a MD4DW [MD06].....	32
Extensiones a MD4DW [MD06].....	33
Desarrollo de la guía.....	35
Esquema de la solución propuesta.....	36
DESCRIPCIÓN DETALLADA Y ANÁLISIS DE LA SOLUCIÓN.....	38
Primera ejecución del caso de estudio.....	38
Segunda ejecución del caso de estudio.....	39
Tercera ejecución del caso de estudio.....	44
Fases 1 y 2.....	44
Fase 3.....	44
Fase 4.....	47
Fase 5.....	49
CONCLUSIONES.....	54
Trabajos futuros.....	55
GLOSARIO.....	56
ANEXOS.....	58
Análisis del contexto de trabajo - Caso de Estudio.....	58
Aplicación de la metodología MD4DW.....	58
Primera ejecución del Caso de Estudio.....	58
Fase 1 - Entender y documentar la realidad.....	58
Fase 2 - Modelado del dominio.....	61
Fase 3 - Construcción del modelo conceptual.....	63
Segunda ejecución – Ontología sin concepto Clave	73
Fase 1 - Entender y documentar la realidad.....	74
Fase 2 - Modelado del dominio.....	74
Fase 3 - Construcción del modelo conceptual.....	75
Aplicación de la guía unificada (Caso de estudio cuentas fondos).....	82
Aplicación de la guía para la construcción de un DW.....	83
Fase 1 - Entender y documentar la realidad.....	83
Fase 2 - Modelado del dominio – Ontología sin modelar claves ni fechas.....	83
Fase 3 - Construcción del modelo conceptual.....	85
Fase 4 - Construcción del ODS (algoritmo ONTO2RDB).....	90
Fase 5 - Generación del diseño lógico y la carga de datos (Algoritmo propuesto por Larrañaga).....	95
E.1 Esquema conceptual.....	95

E.2 Base de datos fuente.....	98
E.3 Lineamientos.....	101
E.4 Correspondencias.....	103
E.5 Aplicación del Algoritmo.....	107
Comparación de resultados.....	120
Conclusiones del caso de estudio.....	122
Metadata de conceptos.....	123
Cuenta.....	123
Dependiente.....	124
Documento.....	125
Estado.....	125
Fondo.....	125
Inversión.....	126
Moneda.....	127
País.....	127
Persona.....	128
Plan.....	128
Portafolio.....	129
Productor.....	130
Región.....	130
Rol.....	131
Algoritmo Onto2rdb.....	132
Precondiciones del algoritmo.....	132
Restricciones.....	132
Algoritmo.....	132
Nuevo esquema.....	132
Una tabla por cada clase.....	132
Claves, relaciones 1 a 1 y relaciones n a 1.....	132
Relaciones n a n.....	134
Jerarquías.....	135
Log de restricciones no consideradas.....	136
Guía Resumida.....	137
Fase 1 - Entender y documentar la realidad.....	138
Fase 2 - Modelado y formalización del dominio.....	138
Fase 3 - Construcción del modelo conceptual.....	140
Fase 4 - Construcción del ODS.....	146
Algoritmo.....	147
Nuevo esquema.....	147
Una tabla por cada clase.....	147
Claves, relaciones 1 a 1 y relaciones n a 1.....	147
Relaciones n a n.....	149
Jerarquías.....	149
Log de restricciones no consideradas.....	149
Fase 5 - Generación del diseño lógico y la carga de datos.....	150
REFERENCIAS.....	154

ÍNDICE DE FIGURAS

FIGURA 1. LA ESTRUCTURA BÁSICA EN EL MODELO MULTIDIMENSIONAL: EL CUBO...	18
FIGURA 2. ESQUEMA E INSTANCIA DE UN CUADRO DE DOBLE ENTRADA.....	18
FIGURA 3. DIMENSIÓN SIN JERARQUÍAS ALTERNATIVAS.....	19
FIGURA 4. DIMENSIÓN CON JERARQUÍAS ALTERNATIVAS.....	19
FIGURA 5. ESPACIO DE CUBOS GENERADO MEDIANTE OPERACIONES DRILLUP.....	20
FIGURA 6. METODOLOGÍA MD4DW.....	22
FIGURA 7. NOTACIÓN GRÁFICA PARA UN ESQUEMA DE NIVEL.....	26
FIGURA 8. NOTACIÓN GRÁFICA PARA UNA DIMENSIÓN.....	26
FIGURA 9. NOTACIÓN GRÁFICA PARA UNA RELACIÓN DIMENSIONAL.....	27
FIGURA 10. TRABAJOS DE CSI RELACIONADOS CON DISEÑO DE DW.....	31
FIGURA 11. FASES DE LA GUÍA DE CONSTRUCCIÓN DE UN DW.....	37
FIGURA 12. REPRESENTACIÓN DE CLAVES	38
FIGURA 13. COMPARACIÓN DE ONTOLOGÍAS.....	41
FIGURA 14. REGLA X MODIFICADA.....	42
FIGURA 15. RESULTADO REGLA XI - SEGUNDA EJECUCIÓN.....	43
FIGURA 16. ONTOLOGÍA - TERCERA EJECUCIÓN.....	44
FIGURA 17. REGLA X - COMPARACIÓN DE RESULTADOS.....	45
FIGURA 18. REGLA X: DIMENSIONES - TERCERA EJECUCIÓN.....	46
FIGURA 19. DIMENSIÓN TIEMPO – TERCERA EJECUCIÓN.....	46
FIGURA 20. TERCERA EJECUCIÓN - RELACIÓN DIMENSIONAL ACTIVOS.....	47

FIGURA 21. COMPARACIÓN ODS - ONTOLOGÍA	48
FIGURA 22. CUBO CAPITAL PORTAFOLIO CON FRAGMENTACIÓN.....	50
FIGURA 23. TERCERA EJECUCIÓN - RESULTADO FASE 5.....	52
FIGURA 24. MD4DW - FASE 1.....	58
FIGURA 25. MD4DW - FASE 2.....	61
FIGURA 26. MD4DW - ONTOLOGÍA.....	62
FIGURA 27. REPRESENTACIÓN DE LAS CLAVES.....	62
FIGURA 28. MD4DW - FASE 3.....	63
FIGURA 29. REGLA I: CLASES CANDIDATAS A NIVELES.....	64
FIGURA 30. REGLA II: CONJUNTO DE NIVELES CON ATRIBUTOS.....	65
FIGURA 31. REGLA IV: PRIMER ORDEN ENTRE NIVELES.....	67
FIGURA 32. REGLA V: NUEVO ORDEN ENTRE NIVELES.....	68
FIGURA 33. REGLA VII: TERCER REGLA DE ORDEN ENTRE NIVELES.....	69
FIGURA 34. REGLA IX: CATEGORIZACIÓN DE SUBCLASES.....	70
FIGURA 35. REGLA X: DIMENSIONES.....	71
FIGURA 36. REGLA XI: RELACIÓN DIMENSIONAL.....	72
FIGURA 37. DIMENSIÓN TIEMPO.....	72
FIGURA 38. DIMENSIÓN FONDO.....	73
FIGURA 39. DIMENSIÓN PORTAFOLIO.....	73
FIGURA 40. RELACIÓN DIMENSIONAL ACTIVOS.....	73
FIGURA 41. ONTOLOGÍA SIN CONCEPTOS DE CLAVES.....	74
FIGURA 42. REGLA I: CLASES CANDIDATAS A NIVELES.....	76

FIGURA 43. REGLA II: CONJUNTO DE NIVELES CON ATRIBUTOS.....	76
FIGURA 44. REGLA IV: PRIMER ORDEN ENTRE NIVELES.....	78
FIGURA 45. REGLA V: NUEVO ORDEN ENTRE NIVELES.....	78
FIGURA 46. REGLA X: DIMENSIONES.....	79
FIGURA 47. REGLA XI: RELACIÓN DIMENSIONAL.....	80
FIGURA 48. DIMENSIÓN TIEMPO.....	80
FIGURA 49. DIMENSIÓN FONDO.....	81
FIGURA 50. DIMENSIÓN PORTAFOLIO.....	81
FIGURA 51. RELACIÓN DIMENSIONAL ACTIVOS.....	81
FIGURA 52. FASES DE LA GUÍA COMPLETA PARA LA CONSTRUCCIÓN DE UN DW.....	82
FIGURA 53. FASE 1.....	83
FIGURA 54. FASE 2.....	83
FIGURA 55. ONTOLOGÍA OBTENIDA EN EL CASO DE ESTUDIO.....	85
FIGURA 56. FASE 3.....	85
FIGURA 57. REGLA I: CLASES A MAPEARSE COMO NIVELES.....	85
FIGURA 58. CLASES A MAPEARSE COMO NIVELES.....	86
FIGURA 59. REGLA IV: PRIMER ORDEN ENTRE NIVELES.....	87
FIGURA 60. REGLA V: NUEVO ORDEN ENTRE NIVELES.....	87
FIGURA 61. REGLA X: DIMENSIONES EN CMDM.....	88
FIGURA 62. REGLA XI: RELACIÓN DIMENSIONAL DEL CASO DE ESTUDIO.....	88
FIGURA 63. DIMENSIÓN FONDO.....	89
FIGURA 64. DIMENSIÓN PORTAFOLIO.....	89

FIGURA 65. DIMENSIÓN TIEMPO.....	90
FIGURA 66. RELACIÓN DIMENSIONAL ACTIVOS.....	90
FIGURA 67. FASE 4.....	90
FIGURA 68. ODS – RESULTADO DE ONTO2RDB.....	94
FIGURA 69. FASE 5.....	95
FIGURA 70. DIMENSIONES Y NIVELES ORIGINALES DEL CASO DE ESTUDIO.....	96
FIGURA 71. DIMENSIÓN PORTAFOLIO CON CLAVES.....	97
FIGURA 72. DIMENSIONES TIEMPO Y FONDO CON CLAVE DEL CASO DE ESTUDIO.....	97
FIGURA 73. RELACIÓN DIMENSIONAL DEL CASO DE ESTUDIO.....	98
FIGURA 74. LINKS UTILIZADOS EN LA BDF.....	100
FIGURA 75. CUBO CAPITAL FECHA.....	101
FIGURA 76. CUBO CAPITAL FONDO.....	101
FIGURA 77. CUBO CAPITAL PORTAFOLIO.....	101
FIGURA 78. CUBO CAPITAL PORTAFOLIO CON FRAGMENTACIÓN.....	102
FIGURA 79. FRAGMENTACIÓN VERTICAL.....	102
FIGURA 80. CORRESPONDENCIAS DE LA DIMENSIÓN TIEMPO.....	103
FIGURA 81. CORRESPONDENCIAS DE LA DIMENSIÓN FONDO.....	103
FIGURA 82. CORRESPONDENCIAS DE LA DIMENSIÓN PORTAFOLIO.....	104
FIGURA 83. CORRESPONDENCIAS DEL CUBO CAPITALFONDO.....	105
FIGURA 84. CORRESPONDENCIAS DEL CUBO CAPITALPORTAFOLIO.....	105
FIGURA 85. CORRESPONDENCIAS DEL CUBO CAPITALFECHA.....	106
FIGURA 86. ESQUEMA LÓGICO RESULTADO.....	118

FIGURA 87. REPRESENTACIÓN RELACIONAL DE LA.....	133
FIGURA 88. RELACIÓN N:1 EN LA ONTOLOGÍA LA ONTOLOGÍA.....	RELACIÓN N:1 EN 133
RELACIONAL DE LA.....	FIGURA 89. REPRESENTACIÓN 134
FIGURA 90. RELACIÓN 1:1 EN LA ONTOLOGÍA DE LA ONTOLOGÍA.....	RELACIÓN 1:1 134
RELACIONAL DE LA	FIGURA 91. REPRESENTACIÓN 135
FIGURA 92. RELACIÓN N:N EN LA ONTOLOGÍA LA ONTOLOGÍA.....	RELACIÓN N:N EN 135
RELACIONAL DE LA	FIGURA 93. REPRESENTACIÓN 136
FIGURA 94. JERARQUÍA EN LA ONTOLOGÍA EN LA ONTOLOGÍA.....	JERARQUÍA 136
FIGURA 95. FASES DE LA GUÍA DE CONSTRUCCIÓN DE UN DW.....	137
FIGURA 96. MATRIZ DE REQUERIMIENTOS.....	139

Introducción

Este proyecto forma parte de una serie de investigaciones relacionadas a proyectos de diseño de DataWarehouse (DW) realizadas por el grupo Concepción de Sistemas de Información (CSI) de la Facultad de Ingeniería de la Universidad de la República Oriental del Uruguay.

Motivación

Un DW es un sistema que almacena datos con el objetivo de convertirlos en información la cual es utilizada como apoyo en la toma de decisiones. Dado que los datos suelen ser generados por más de un sistema operacional, una dificultad que generalmente se enfrenta es la heterogeneidad de las fuentes y datos. O sea, los diferentes tipos de fuentes de datos que pueden utilizarse (xml, bases relacionales, archivos propietarios, etc) y las múltiples representaciones para un mismo dato que pueden existir en ellas. La heterogeneidad de las fuentes de datos implica definir un subconjunto de la información que sea relevante para los propósitos del DW y realizar los programas de carga que nutren de información al nuevo sistema.

Los diversos proyectos realizados dentro de CSI se ocupan de diferentes etapas del proceso de desarrollo de un DW y cada uno de ellos parte de un supuesto que restringe la utilización del mismo y no siempre coincide con el resultado del proyecto correspondiente a la etapa anterior. Este trabajo pretende realizar la unificación de dichos proyectos de manera de obtener una guía que contempla el desarrollo de un DW desde su inicio hasta su diseño lógico. Para esto, se utilizarán los trabajos existentes para cubrir cada etapa y se agregarán guías para las etapas que no han sido contempladas o para realizar la unión entre dos trabajos, transformando la salida de uno en la entrada del siguiente.

Hasta ahora en el mencionado grupo se trató el problema de diseño del DW y la carga del mismo con una única fuente, asumiendo que esta única fuente tiene todos los datos consolidados que son relevantes para el DW. Este trabajo presenta una aproximación a una solución para este problema que se basa en la generación de un Operational Data Store (ODS) y además propone un diseño de carga del DW desde dicha base. La solución es parcial porque no se soluciona la carga de cada una de las fuentes a dicho ODS.

Objetivos

Obtener una guía que abarque una mayor cantidad de etapas del proceso de desarrollo de un DW que las presentadas en el proyecto de grado MD4DW [MD06], utilizando proyectos relacionados a la temática y desarrollados por CSI.

Detectar fortalezas y debilidades de los proyectos en los que se basa, para obtener el mayor provecho de los puntos fuertes y encontrar posibles soluciones a los débiles.

Permitir la carga desde varias y heterogéneas fuentes de datos, punto en el cual no se llega a un resultado completamente satisfactorio, pero sí se realiza un avance en procura de una guía completa para ello.

Medir la eficacia y eficiencia de la guía, aplicando la misma a un caso de estudio real y comparando resultados con un equipo que no la utiliza.

Caso de estudio

Como se menciona anteriormente, el resultado de este trabajo es aplicado a un caso de estudio real en la empresa uruguaya AIVA [AIVA], la cual se especializa en carteras de inversión en todo Latinoamérica.

La empresa planteó dos necesidades que se presentaron en el departamento de Asset Management de la misma y que su equipo de IT se disponía a desarrollar. Dichas necesidades fueron relevadas tanto por el equipo interno de la empresa como por el equipo que presenta este trabajo.

Una vez diseñado el DW, se prosiguió a la comparación de resultados tanto en términos de tiempo como en correctitud del mismo. Aunque el equipo interno de la empresa contaba con la ventaja de conocer el negocio, los resultados finales fueron satisfactorios en cuanto a las horas empleadas como así también en el resultado final del diseño.

Organización del documento

Además de la introducción presentada en este primer capítulo, este trabajo cuenta con 7 capítulos más.

En el capítulo 2 se presenta una reseña del proceso de construcción de un DW, junto a los problemas que pueden acontecer en cada etapa y los riesgos a los que se puede enfrentar un equipo de desarrollo.

En el capítulo 3 se realiza un resumen de los proyectos estudiados para realizar este trabajo, junto a un análisis de sus fortalezas y debilidades.

El capítulo 4 muestra los caminos explorados para cumplir con los objetivos planteados en este proyecto y el camino definitivo escogido.

En el capítulo 5 se detalla y analiza la solución a cada tarea necesaria para obtener el DW y a cada problema encontrado al momento de ejecutar el caso de estudio.

El capítulo 6 presenta las conclusiones finales del trabajo realizado y por último, los capítulos 7 y 8 contienen el Glosario y los anexos respectivamente.

Reseña del proceso de construcción del DW

Concepto de DataWarehouse

Un DataWarehouse (DW) es una base de datos cuyo objetivo principal es dar soporte a la toma de decisiones ([RE06]). Para esto es necesaria gran cantidad de información, la cual es adquirida desde una o más fuentes de datos (bases de datos, archivos u otros sistemas de almacenamiento de datos), siendo una fuente extremadamente importante los sistemas operacionales.

El DW debe homogeneizar la información que se almacena en él desde diversas fuentes, para lo cual suelen usarse ODS.

Por su cometido este tipo de base de datos debe responder un gran número de consultas complejas y sufre muy pocas actualizaciones, por lo cual criterios de diseño de base de datos como la normalización no aplican de la misma forma que en las tradicionales bases de datos transaccionales.

Un DW cumple con las siguientes características:

- Integrado: Se pueden tener datos de muchas fuentes, pero deben eliminarse las inconsistencias.
- Histórico: El tiempo es siempre parte del análisis en la toma de una decisión (generalmente cuando se piensa en el futuro, se analiza el pasado para sacar conclusiones) por tanto siempre estará presente el tiempo en un sistema de este tipo. Los sistemas operacionales tienden a dar una imagen del estado actual, una “foto actual”, mientras un DW tiende a presentar un conjunto de fotos conformando un “video” o una “secuencia de diapositivas”.
- No volátil: A diferencia de las bases transaccionales en las que la información se actualiza en tiempo real, en un DW se mantiene información estable. Es decir que en un DW las actualizaciones son definidas mediante procesos específicos habitualmente incrementales. En la mayoría de los casos no suele borrarse o modificarse, sino agregar nueva información. Siguiendo con la analogía del punto anterior, lo que se hace es agregar nuevas imágenes y no borrar o cambiar las que ya hay para seguir construyendo el mencionado “video”.

Dadas estas características se han generado también modelos conceptuales más acordes que los convencionales Modelo Entidad Relación (MER), como ser CMDM el cual es descrito en el contexto de trabajo de este documento.

Como se plantea en el proyecto de grado “Metadata for Datawarehouse” (MD4DW) [MD06], el proyecto de desarrollo de un sistema de DW, al igual que todo desarrollo en el área IT, tiene las siguientes etapas:

- Análisis
- Diseño
 - Diseño conceptual
 - Diseño lógico
 - Diseño físico (implementación del diseño lógico)
- Implantación

Análisis

En esta etapa se intenta entender la realidad y los objetivos del sistema; para llevar a cabo esto, se requiere de gran interacción con expertos en la realidad de la organización. Finalizada esta etapa algunos de los integrantes del grupo de desarrollo (analistas) deben tener un sólido conocimiento de la realidad, conceptos, terminología y las reglas del negocio, así como también deben conocer las necesidades (requerimientos). Una vez que se tiene el suficiente conocimiento sobre la realidad y los requerimientos es posible pasar a la etapa de diseño.

Diseño

Esta etapa se divide en tres sub-etapas mediante las cuales se materializa la solución.

Diseño conceptual

En esta sub-etapa se realiza la transformación de la especificación formal de los requerimientos en un modelo formal (esquema conceptual multidimensional).

Diseño lógico

En esta sub-etapa se toma como entrada la salida de la sub-etapa anterior y se crea un diseño lógico que puede ser relacional o multidimensional. El objetivo es encontrar un diseño que contemple todos los requerimientos incluyendo los de rendimiento en las consultas más complejas.

Diseño físico

En esta sub-etapa se implementa el diseño obtenido en el punto anterior en un manejador de base de datos, teniendo en cuenta todos los detalles de implementación del manejador seleccionado, como índices, particiones y lo que pueda aportarnos la base de datos seleccionada.

Implantación

En esta etapa el objetivo es poner a funcionar el sistema diseñado previamente. Para ello debe llevarse a cabo la instalación del mismo y realizarse los procedimientos de **carga de datos** desde las diversas fuentes.

Carga de datos

Se define como carga al proceso de llenar el DW con la información adecuada; este proceso conocido como ETL por sus iniciales en inglés incluye tres partes:

Extract

Es el proceso de conseguir la información desde la(s) fuentes de datos

Transform

Se define como el proceso de transformar, homogeneizar y limpiar los datos extraídos para así poder cargarlos en el DW

Load

Es el proceso de cargar los datos transformados en el DW.

Esta suele ser una tarea difícil y costosa debido a la diversidad de fuentes de datos, y por tratarse de la parte final del desarrollo que es la época que típicamente tiene más estrés en todo el proyecto.

Riesgos y problemas del proceso de construcción

Profundizando un poco más en el proceso de construcción de Datawarehouse antes descrito se puede añadir que cada una de sus etapas presenta riesgos y problemas. A continuación se describen los principales problemas de cada etapa.

La dificultad de la etapa de análisis radica en lograr un consenso entre los analistas y los clientes en la descripción de la realidad, la definición de los conceptos, las reglas de negocio y los actores que en ella participan.

En la etapa de diseño conceptual el problema principal es encontrar un modelo formal que permita especificar la realidad relevada en la etapa de análisis, con sus conceptos, relaciones, restricciones, etc.. Otro riesgo, de gran impacto en las etapas posteriores, es no detectar inconsistencias o ausencia de datos. Su detección en esta etapa es de suma importancia para no tener que volver atrás en las etapas siguientes.

Con respecto a la etapa de diseño lógico, la dificultad se encuentra en identificar exitosamente los roles de los conceptos encontrados; esto es, cuáles serán dimensiones, cuáles serán niveles y cómo estarán formadas las jerarquías. También es un desafío definir un modelo lógico que, además de satisfacer los requerimientos funcionales de información, satisfaga los requerimientos de performance en la realización de consultas complejas de análisis de datos.

Observando ahora el proceso completo, resulta complejo documentar adecuadamente cada etapa, de manera que dicha documentación pueda ser efectivamente validada con el cliente, así como empleada en las etapas siguientes, manteniendo la trazabilidad a lo largo del proceso. Además, si no se mantiene dicha trazabilidad el análisis de impacto de un cambio no podrá realizarse con éxito. Cabe agregar que un proyecto en el cual se deban contemplar múltiples fuentes de datos deberá resolver problemas específicos de dicha realidad, incrementando los riesgos del mismo.

Modelos multidimensionales

Esta clase de modelo suele emplearse en el diseño lógico de Datawarehouse, ya que es más adecuado mantener una visión multidimensional de los datos en sistemas de datawarehousing [KT96].

Definiciones [Car00]:

Estructuras Básicas: Cubos, Dimensiones y Medidas.

En una base de datos multidimensional la información se representa como matrices multidimensionales, cuadros de múltiples entradas o funciones de varias variables sobre conjuntos finitos denominadas “Cubos” (Figura 1).

El **esquema** de un cubo queda determinado dando a conocer sus ejes con sus respectivas estructuras y la estructura de los datos que se presentan en cada celda de la matriz. Se asume que todas las posiciones de la matriz tienen datos con igual estructura (datos uniformes).

A los ejes se les llama **Dimensiones**, al dato que se presenta en la matriz se le llama **Medida** y a los elementos del producto cartesiano de los ejes se le llama **Coordenadas**. La matriz definida puede ser dispersa.

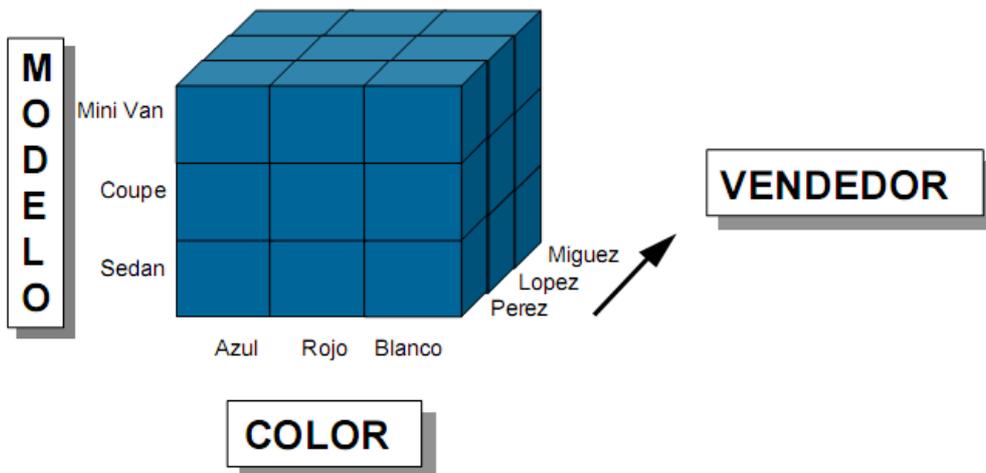


Figura 1. La estructura básica en el modelo multidimensional: el Cubo

Una **instancia** de un cubo, queda determinada por un conjunto de datos para cada eje y un conjunto de datos para la matriz (Figura 2).

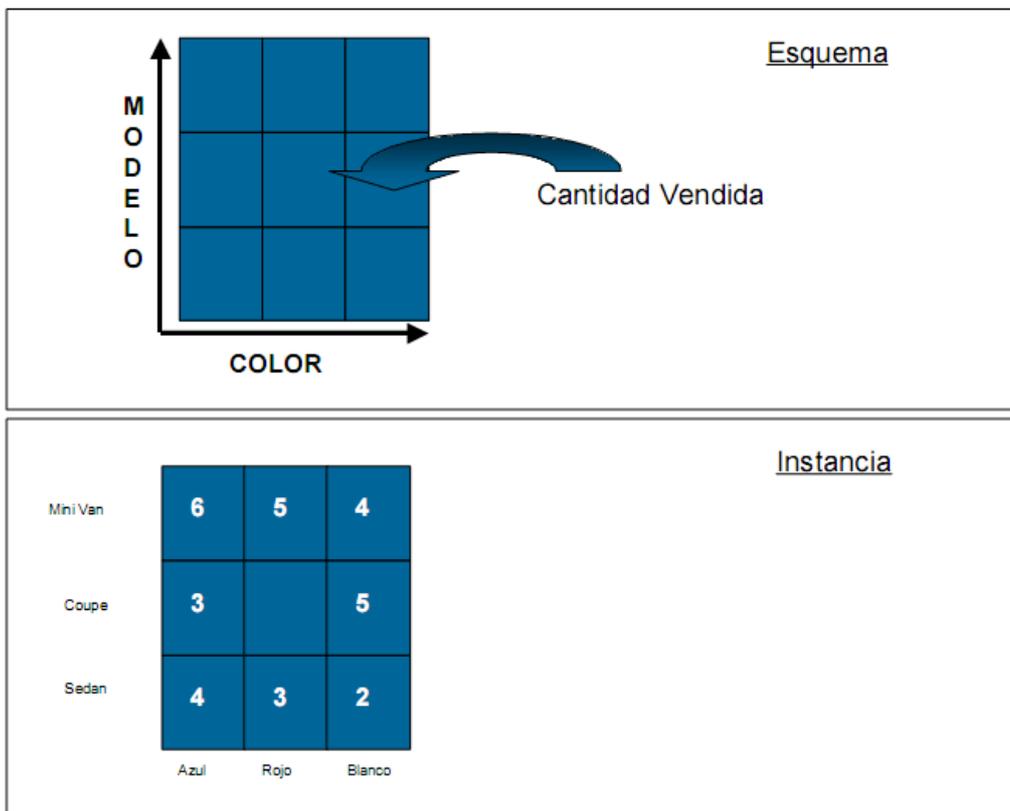


Figura 2. Esquema e instancia de un cuadro de doble entrada

Dimensiones con Jerarquías.

Normalmente las dimensiones se estructuran en jerarquías (Figura 3).

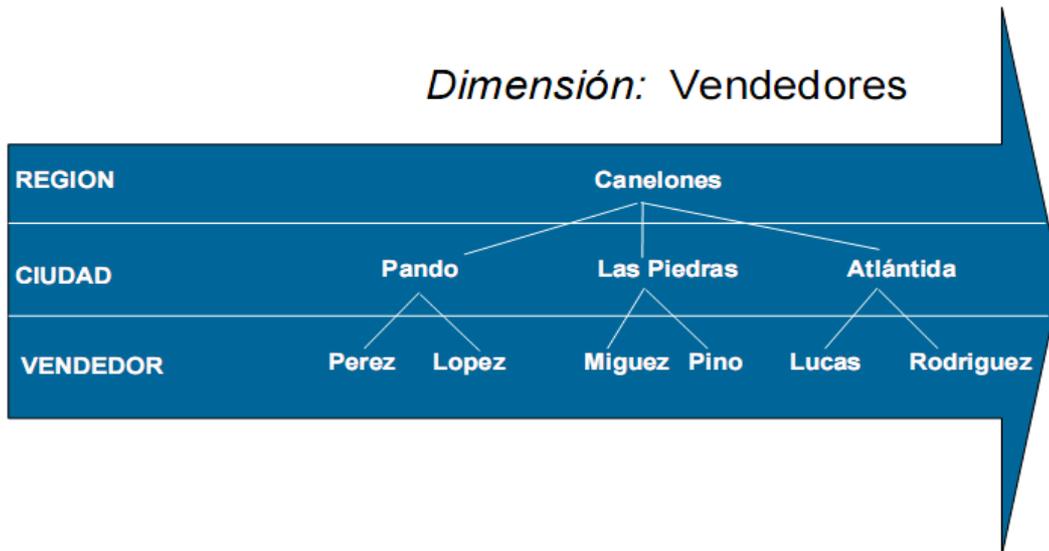


Figura 3. Dimensión sin jerarquías alternativas

A cada nivel de una jerarquía se le llama **Nivel de Agregación** o simplemente Nivel.

Una dimensión puede tener más de una jerarquía (Figura 4).

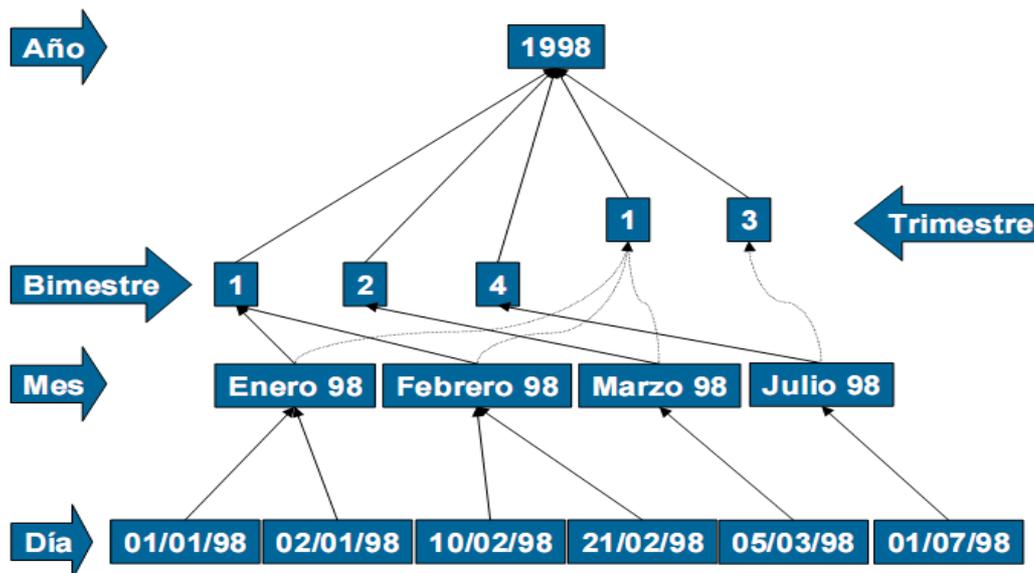


Figura 4. Dimensión con jerarquías alternativas

Operaciones Multidimensionales y Espacios de Cubos.

Las operaciones multidimensionales se pueden agrupar en tres conjuntos básicos:

1. *De selección y visualización o Slice & Dice.*

- Slice: Selecciona “dimensiones de trabajo” de un cubo mayor.
- Dice o Filtrado: Selecciona “secciones” del cubo en función de valores de las dimensiones.
- Rotación: Permite “presentar” diferentes planos de un cubo.

2. *De Agregación:* realizar “movimientos” en las jerarquías de las dimensiones.
 - a. DrillUp: Se “sube” de nivel por una jerarquía agrupando todos los valores del nivel original que están relacionados con el mismo valor del nivel superior.
 - b. DrillDown: Se “baja” por la jerarquía produciendo la desagregación de dichos valores.
 - c. Roll-up o Consolidación: Calcular una nueva medida en función del conjunto de los valores de las medidas que se agrupan al realizar DrillUp. Esto se traduce típicamente en funciones de agregación como las presentes en SQL (sum, avg, etc.)

3. *De Relacionamiento:* Acceder a otros datos.
 - a. Drill-Across: Los datos a acceder están en un cubo.
 - b. Drill-Through: Los datos están en el DataWarehouse o en la base operacional.

Dado un cubo, al aplicar operaciones de DrillUp o DrillDown, se recorre un “espacio de cubos”. Dicho espacio está determinado por las dimensiones que participan en el cubo origen y la forma en que se deben realizar los cálculos con las medidas (RollUp) en cada DrillUp (Figura 5).

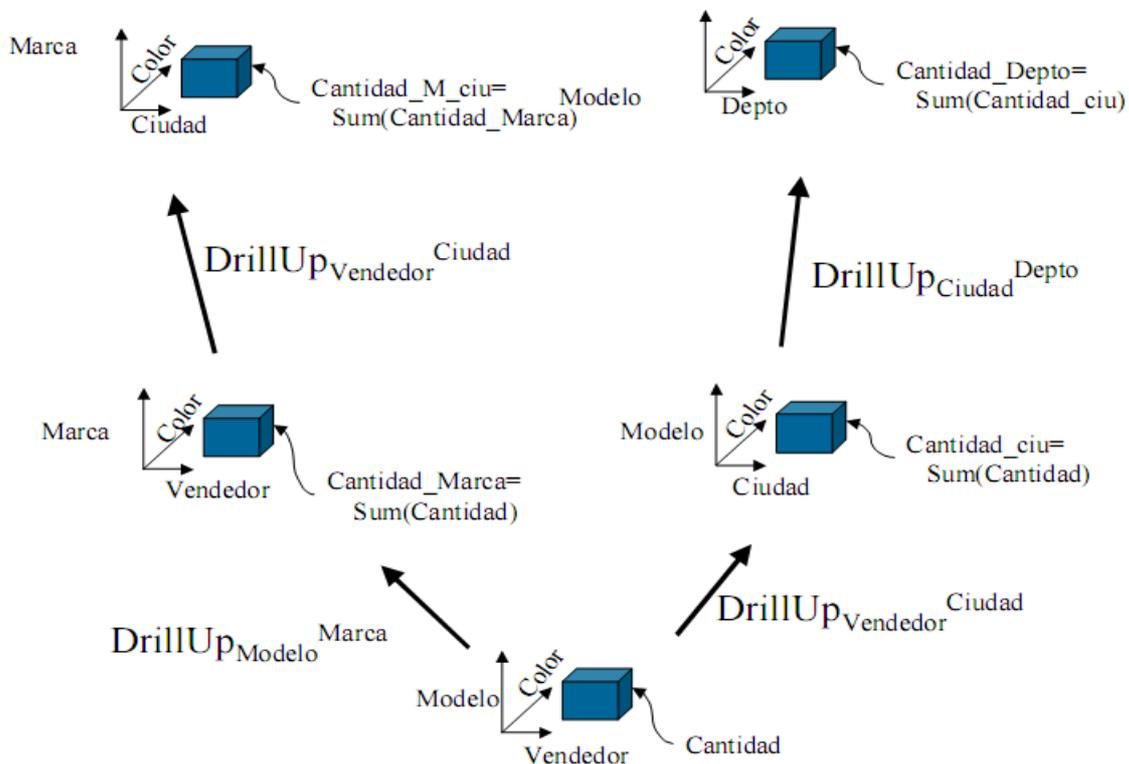


Figura 5. Espacio de cubos generado mediante operaciones DrillUp

Rol de la metadata en el proceso de DW

En un proceso de Datawarehouse, además de conocer el origen de datos y los datos en sí, es necesario tener un conocimiento de la semántica de los mismos y de las relaciones que entre ellos existen ([MJ97]). Para esto, la metadata es una herramienta que permite registrar el conocimiento de los datos, el cual será fuente de información para otros y para un posterior mantenimiento del Datawarehouse. Además, en la metadata se encuentran definiciones y significados de carácter semántico que son de interés para las personas que toman decisiones en función de la información que alberga el Datawarehouse. Se puede considerar la metadata

como un elemento más dentro de la solución a un proyecto de Datawarehouse, dado que desde el punto de vista conceptual brinda información de los conceptos que forman parte de la realidad sobre la cual se establece el Datawarehouse. Desde el punto de vista técnico es un elemento de apoyo a lo largo del proyecto ya que administra información empleada por los desarrolladores que definen la carga del Datawarehouse.

La metadata enriquece un Datawarehouse y facilita el proceso de construcción y mantenimiento del mismo.

Contexto de trabajo

Ante la gran variedad de problemas y riesgos que pueden presentarse en un proyecto de construcción de Datawarehouse, tal como se describió en el capítulo anterior, se decidió relevar cuáles habían sido las propuestas de soluciones de apoyo a dichos proyectos. El estudio se focalizó principalmente en las propuestas presentadas dentro del grupo Concepción de Sistemas de Información (CSI) del Instituto de Computación, dado que este trabajo surgió del proyecto MD4DW [MD06] propuesto dentro del mismo; algunos autores externos a dicho entorno también fueron abarcados para relevar propuestas a necesidades específicas que se encontraron dentro del proceso. A continuación se describen los trabajos relevados.

MD4DW

La metodología propuesta por MD4DW [MD06] brinda una guía general para el diseño del modelo conceptual de un Datawarehouse representado mediante un modelo multidimensional CMDM [Car00].

Esta metodología se destaca por seguir etapas o fases justificadas a través de las cuales de forma guiada se consideran, además de las necesidades iniciales, el mantenimiento y futuros cambios del Datawarehouse. Estos últimos dos puntos son considerados a través de la documentación generada a lo largo de la aplicación de la metodología, la cual permite mantener una gestión del conocimiento relativo al dominio y de las decisiones tomadas durante el diseño del Datawarehouse.

El ciclo de vida de la metodología MD4DW [MD06] consiste en tres fases secuenciales, ya que la salida de cada una es la entrada de la fase siguiente.

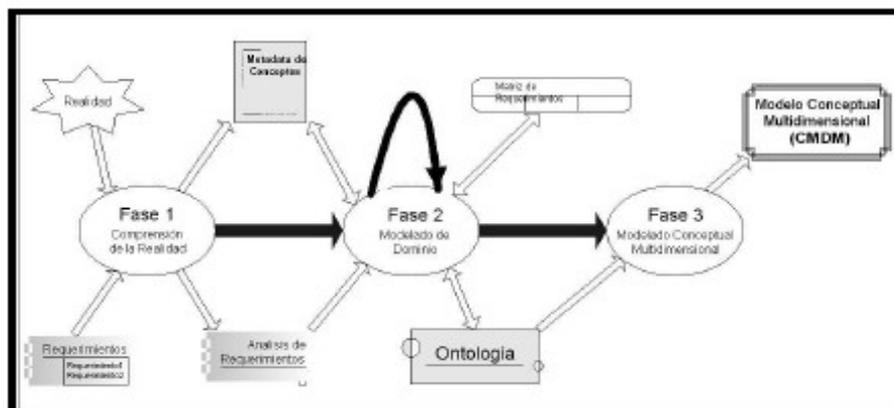


Figura 6. Metodología MD4DW

Aunque no está contemplado en el flujo de la Figura 6, el proceso se puede repetir tantas veces como sea necesario, siempre teniendo en cuenta que para pasar a la siguiente fase se debe realizar una validación del resultado de la fase actual para evitar reflujos innecesarios.

Descripción de las Fases

Fase 1: Comprensión de la realidad

En esta primera fase se propone realizar un análisis profundo de la realidad en la cual se diseñará el Datawarehouse. Este análisis consiste en entender completamente el dominio en el cual se está trabajando, así como la terminología usada dentro del mismo, de forma que no existan ambigüedades conceptuales que puedan llevar a una solución errónea a los requerimientos iniciales.

El punto de entrada de esta fase es el conjunto de requerimientos iniciales. En base a ellos se debe determinar el dominio dentro del cual serán desarrollados. La información relevada sobre el dominio debe documentarse en forma concisa y clara en el documento Metadata de Conceptos. En dicho documento se debe describir el dominio general de análisis y las características para cada concepto participante del mismo.

Fase 2: Modelado de Dominio

Una vez comprendida la realidad en la cual se está trabajando, y en base al documento de Metadata de Conceptos, se debe generar un modelo formal que represente los conceptos relevados junto a sus relaciones y restricciones.

A través de este modelo formal se abstrae la realidad descrita anteriormente mediante lenguaje natural, obteniendo un modelo semántico que represente el dominio en el cual se eliminaron las posibles inconsistencias y/o conflictos que pudieran surgir. Los cambios realizados en este modelado deberán reflejarse en el documento Metadata de Conceptos logrando así mantener consistencia y trazabilidad en la documentación a lo largo del proceso.

En esta fase se propone definir una matriz de requerimientos, la cual permitirá medir el impacto de los cambios efectuados en los requerimientos analizados.

La metodología propone representar el modelo de dominio a través de una ontología, la cual permite trabajar sobre un lenguaje formal y el uso de razonadores para verificar la coherencia y la consistencia de los conceptos, relaciones y restricciones definidas.

Fase 3: Modelado Conceptual Multidimensional

Una vez alcanzados los objetivos de las fases anteriores, se cuenta con el documento Metadata de Conceptos, la Matriz de Requerimientos y el modelo de dominio referentes a los requerimientos iniciales relevados. En base a estos documentos y mediante la aplicación de un conjunto de reglas definidas en la metodología, se transformarán los conceptos y relaciones representados en el modelo de dominio en estructuras del modelo conceptual multidimensional.

Las once reglas propuestas se pueden agrupar en tres grandes conjuntos: (1) Definición de niveles (2) Definición de Jerarquías y (3) Definición de Dimensiones y Relaciones Dimensionales.

Al finalizar esta fase, se debe contar con un modelo conceptual multidimensional CMDM [Car00].

Fortalezas y debilidades

Fortalezas

- Esta metodología incluye análisis de requerimientos y en base a ellos realiza el diseño de la solución.
- Usa metadata, lo cual aporta valor agregado al Datawarehouse (ver Sección)
- Brinda documentación a lo largo del proceso, dejando documentado desde el análisis de dominio hasta las decisiones de diseño.
- Permite una validación directa y clara con el cliente a través del documento Metadata de Conceptos y la ontología.

Debilidades

- Algunas reglas de la metodología no fueron fácilmente comprensibles.
- No define claramente dónde y cómo especificar los identificadores de conceptos en la Metadata propuesta.
- No propone forma de representación de claves en la ontología.
- Solo cubre análisis y diseño conceptual, sin llegar al diseño lógico y la carga.
- No considera el(los) origen(es) de dato(s)

OWL2DSS

El proyecto “Diseño de un DSS a partir de metadata expresada en OWL” [DSS06] parte de un generador de decision Support System (DSS) desarrollado en el INCO a partir de metadata heterogénea. Dicho generador tiene una arquitectura denominada DSSMet, la cual es altamente acoplada, lo que dificulta la reutilización de componentes y su crecimiento. Presenta además, heterogeneidad de lenguajes de modelado, entre ellos DAML+OIL [DAML], MER y XML.

Aprovechando el know-how adquirido en este sistema y motivados por el advenimiento de las ontologías, han comenzado esfuerzos para transformar el generador actual en un generador de DSS basado puramente en ontologías OWL, buscando que este generador tenga una arquitectura flexible y desacoplada.

Se decide utilizar ontologías debido a que las organizaciones especifican sus dominios mediante éstas y es cada vez más viable la interoperabilidad de sistemas que mantengan vinculaciones sobre la misma metadata. Por otro lado, las herramientas de razonamiento sobre ontologías son cada vez más poderosas, por lo que no se descarta la ganancia adicional de poder aplicarlas sobre los diferentes modelos que genere el sistema.

En este proyecto se explora la arquitectura a alto nivel de un generador de DSS y se subdivide el proceso de generación del mismo en varios subprocesos que utilicen ontologías OWL, tanto de entrada como salida. Asimismo, presenta la implementación de parte de esta arquitectura: un plug-in de Protégé [Prt] para generar modelos multidimensionales CMDM a partir de ontologías OWL, basándose en el proyecto de F.Carpani [Car00].

Fortalezas y debilidades

Fortalezas

- Basándose en la arquitectura existente DSSMet se diseña una nueva arquitectura que permite el uso de ontologías en la generación de un DSS.

- Se divide el proceso general en subprocesos más simples de implementar y se centra el proyecto en sólo uno de estos subprocesos (OWL2CMDM), pero se dejan pautas para avanzar en el resto.

Debilidades

- No se representan en OWL las restricciones mencionadas de CMDM. Si bien se proporcionó una solución genérica para tratarlas, no se proporcionó una solución para cada una de las restricciones mencionadas en la tesis de F. Carpani [Car00]
- La posibilidad del usuario de expresar sus elecciones (requerimientos) no fue resuelta debido a que se concentraron los esfuerzos en otros aspectos del proyecto. Una noción clara de los requerimientos facilitaría la construcción de plantillas para generar el DSS automáticamente.
- El plugin no recuerda la posición de los elementos del CMDM, por lo que al cerrar la aplicación se pierde el orden de los mismos.
- Las reglas y estrategias no están relacionadas, por lo que puede darse el caso de que se trabaje con una estrategia que llame a reglas no disponibles.

CMDM

F. Carpani en su tesis de maestría [Car00] propone un lenguaje para representar modelos conceptuales de bases de datos multidimensionales, el cual ofrece un lenguaje gráfico y un mecanismo para especificar restricciones. Con este mecanismo se pueden especificar conjuntos de cubos con características comunes, permitiendo la especificación de un conjunto mayor de opciones de análisis que simplemente las que se den una a una, como se hace tradicionalmente.

El modelo apunta solamente al diseño conceptual, presentando el diseño y prototipo de una herramienta CASE orientada al diseño de DataWarehouses.

Estructuras en CMDM

El objetivo fundamental de CMDM es permitir la especificación de una determinada realidad en términos multidimensionales, soportando al menos las estructuras presentadas anteriormente: Cubos, Dimensiones y Medidas. Para lograr esto CMDM presenta tres estructuras básicas:

1. Niveles.
2. Dimensiones.
3. Relaciones Dimensionales.

Niveles

Un nivel representa un conjunto de objetos que son de un mismo tipo. Cada nivel debe tener un nombre y un tipo. Conceptualmente no tiene diferencia con cualquier elemento del Modelo Entidad Relación que pueda ser considerado un conjunto de entidades (Figura 7).

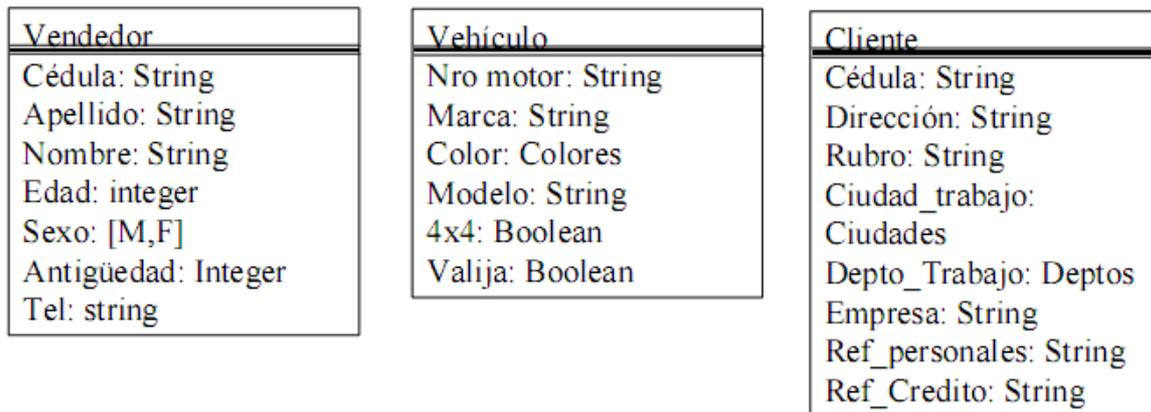


Figura 7. Notación gráfica para un esquema de nivel

Dimensiones

Una dimensión está determinada por una jerarquía de niveles y la instancia es una jerarquía de elementos de esos niveles. El esquema de una dimensión se representa en la Figura 8.

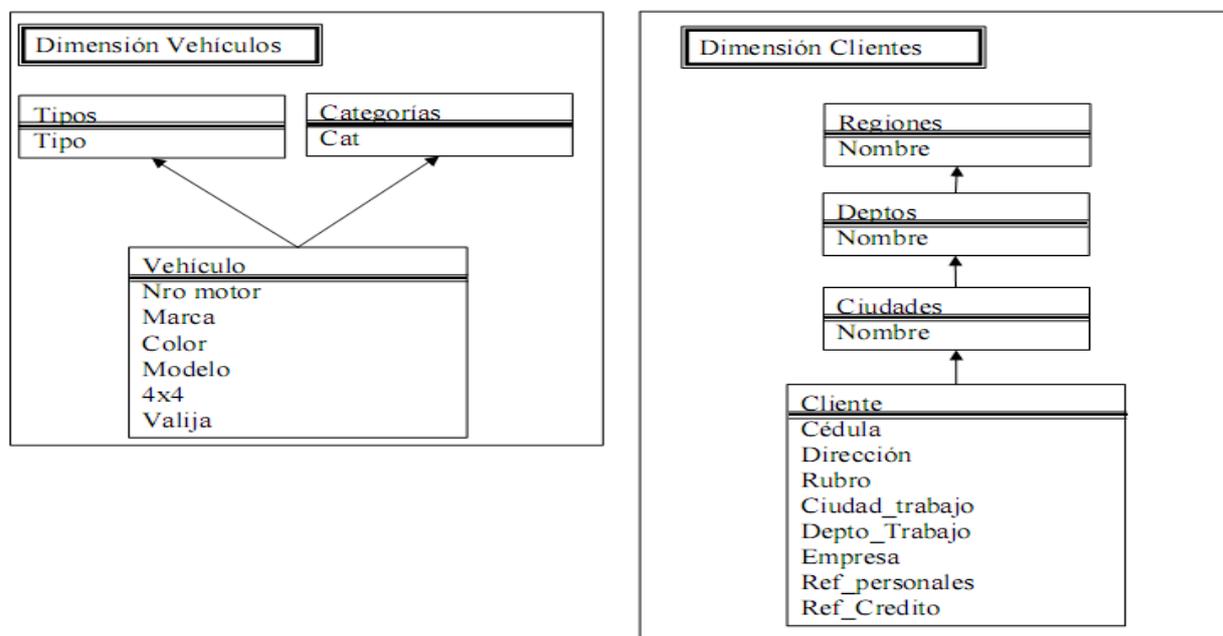


Figura 8. Notación gráfica para una Dimensión

De esta forma, el esquema de una dimensión está representado por un rectángulo dentro del cual aparece un nombre para la dimensión y un grafo dirigido en donde los nodos son los niveles que participan de esa dimensión.

Relaciones Dimensionales

Una relación dimensional representa un conjunto de cubos, tomado del conjunto de todos los cubos que se pueden construir a partir de los niveles de un conjunto dado de dimensiones. El dibujo de la Figura 5 se puede interpretar como una instancia de una relación dimensional, dado que es un conjunto de cubos construido a partir de los niveles de un conjunto dimensiones.

Se asume que en cada uno de los cubos que pertenecen a la instancia de la relación dimensional debe aparecer al menos un nivel de cada una de las dimensiones que participan en la relación.

En CMDM, un cubo es una función que va del producto cartesiano de las instancias de los niveles en los boléanos; de esta forma, cualquier nivel puede cumplir el rol de medida. Notar que, en CMDM, los cubos son elementos de las instancias de las relaciones dimensionales y no hay una estructura que los represente directamente.

Por lo tanto, el esquema de una relación dimensional está dado por un grafo en forma de estrella, donde el nodo central es de forma oval y tiene el nombre de la relación dimensional y los nodos “satélite” son rectangulares y tienen el nombre de cada una de las dimensiones que participan de la relación como se muestra en la Figura 9.

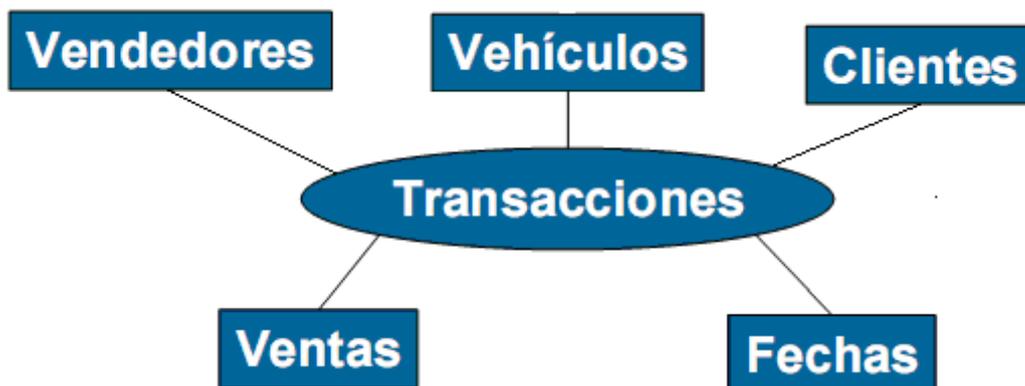


Figura 9. Notación gráfica para una Relación dimensional

Fortalezas y debilidades

Fortalezas

- Este modelo es totalmente independiente del lenguaje que se utilice en la implementación del mismo.
- Es independiente del modelo relacional.
- Cada definición es presentada en un lenguaje formal, lo cual evita ambigüedades en las mismas. Se define un lenguaje gráfico para representar estructuras (Niveles, Dimensiones y Relaciones dimensionales) y restricciones, el cual puede ser utilizado para la comunicación entre las personas involucradas.
- El modelo permite una visión multidimensional de los datos debido a que las estructuras propuestas son multidimensionales. Además soporta dimensionalidad genérica ya permite cubos sin medida y resumibilidad a través de restricciones de integridad.

Debilidades

- No presenta un algoritmo que permita chequear la consistencia de las restricciones.
- El manejador CMDM presenta algunas debilidades en cuanto al manejo gráfico de restricciones y posibilidades de impresión.
- El modelo CMDM es utilizado solo dentro de CSI. Otros modelos estándar podrían tener mayor difusión.
- El lenguaje de restricciones que propone es complejo.

Peralta

V.Peralta en su trabajo “Diseño Lógico de Data Warehouses a partir de esquemas conceptuales multidimensionales” [Per01], presenta un algoritmo para la generación semiautomática de un esquema relacional para un DW a partir de una base de datos fuente integrada, un esquema conceptual modelado con CMDM [Car00] y un conjunto de correspondencias entre la base de datos fuente y el esquema integrado. Este trabajo se basa en una tesis de maestría de Adriana Marotta [Mar00].

El algoritmo produce como resultado un esquema relacional y una Traza de Primitivas también llamada Traza de Diseño (o simplemente Traza).

Fortalezas y debilidades

Fortalezas

- Especifica claramente las entradas necesarias para realizar el diseño lógico de un DW.
- Especifica reglas de diseño y presenta un algoritmo que las ordena.
- Implementa una herramienta CASE para realizar el diseño lógico de un DW.

Debilidades

- No es posible aplicar el algoritmo para cualquier nivel de fragmentación, ya que no permite la fragmentación vertical de los niveles.
- El algoritmo no abarca todas las primitivas, por ejemplo Minidimension Break Off y Data Array Creation, para las cuales sería necesario la creación de nuevos lineamientos e incluso nuevas reglas de diseño.
- Falta agregar criterios de calidad y funciones de costo.
- No contempla todas las restricciones de integridad expresables en CMDM.
- No propone guías para efectuar la fragmentación de los cubos o de las dimensiones.
- No considera la carga del Datawarehouse.
- Se enfoca en el diseño del modelo lógico en base a un modelo CMDM y su origen de datos; no son considerados los contextos con varios orígenes ni cómo se obtuvo el modelo conceptual.
- No propone documentación respecto a la semántica de los conceptos que intervienen en el Datawarehouse.

Larrañaga

I.Larrañaga en su trabajo “Carga de un Datawarehouse a partir de la Traza de Diseño” [Lar06], aborda el problema de la carga y actualización del DW reutilizando el conocimiento generado durante el diseño conceptual y lógico de éste. En particular, se toma como base el algoritmo propuesto por V. Peralta [Per01]. Utilizando la información disponible del algoritmo antes mencionado, este trabajo analiza los resultados que se obtendrían con un **enfoque naive**. Este enfoque considera la sentencia SQL relacionada con cada primitiva y se limita a proponer como solución la secuencia de vistas asociada a la secuencia de primitivas.

I.Larrañaga realiza la ejecución de dicho algoritmo, e identifica y detalla algunos problemas que resuelve con una nueva propuesta del algoritmo.

El principal aporte de I.Larrañaga en [Lar06] es la presentación de una solución al problema de carga y actualización para un DW diseñado utilizando la metodología propuesta por V. Peralta en [Per01].

Este trabajo presenta un algoritmo que permite generar una sentencia SQL para cada relación en el DW, llevando ahora el problema a un contexto donde existe más literatura y posibilidad para trabajos futuros.

Este algoritmo utiliza como entrada la misma información que en su momento se usó para generar el esquema, lo cual significa que es directamente aplicable luego de haber obtenido el diseño del mismo.

Siguiendo una secuencia de transformaciones, aquellas utilizadas para la construcción del esquema, este trabajo se aboca a proponer cómo manipular los datos subyacentes, focalizándose en las transformaciones más que en la extracción y la carga.

Fortalezas y debilidades

Fortalezas

- Presenta vistas unificadas. A diferencia de V. Peralta en [Per01], Larrañaga en este trabajo propone que en cada paso se vaya utilizando el resultado del paso anterior en forma directa, obteniendo así una única vista al final del algoritmo.
- Propone un diseño de la carga del DW.

Debilidades

- Solución basada en SQL, lo cual no permite que sea aplicado en cualquier entorno.
- A pesar de identificar errores de diseño (claves primarias incorrectas, diferencias de tipos, valores nulos, etc.) y errores introducidos en el proceso de carga, este trabajo sólo trata éstos últimos, dejando para futuras etapas el tratamiento de errores de diseño.
- Se enfoca en el diseño de la carga inicial del DW pero no en la actualización del mismo.
- Falta analizar cómo agregar medidas de calidad y el impacto de la evolución de las fuentes de datos.
- Mantiene debilidades encontradas en el trabajo de V. Peralta [Per01]: no propone guías para realizar la fragmentación de cubos y dimensiones, no hay información semántica de los datos, no se consideran varios orígenes de datos.

Bernstein

Este trabajo es uno de los proyectos realizados fuera de CSI, pero se estudió debido a la relevancia del mismo para este trabajo. En el artículo “*A Vision for Management of Complex Models*” [Ber00] se plantea la importancia de definir una gestión de modelos de datos genérica, que sirva para todos los posibles modelos que se puedan encontrar en cualquier escenario.

Un escenario específico que se plantea en dicho artículo, de interés en el presente proyecto, es el mapeo de orígenes de datos en tablas de un datawarehouse, en el cual deben generarse programas que transformen los datos de origen y los carguen en el datawarehouse. Uno de los argumentos de Bernstein [Ber00] es que en lugar de implementar operaciones de manejo de modelos comunes una y otra vez para diferentes tipos de esquemas de datos, se podrían implementar una sola vez operaciones genéricas en un gestor de modelos de datos más abstracto. Pero de todas formas será necesario implementar programas de usuario específicos para casos particulares que interpreten o apliquen determinados mapeos.

Los tipos de aplicaciones de gestión de modelos planteados en este artículo son, en general, considerados ejemplos de “gestión de Metadata”, donde el mayor esfuerzo en la construcción de la aplicación se encuentra en la manipulación de “descripciones” de los objetos de interés, en lugar de los objetos en sí mismos.

Uno de los objetivos fundamentales en la gestión de modelos es el desarrollo de un mecanismo de representación de modelos y de almacenamiento de estas representaciones. Un problema clave aquí es cuanta semántica del modelo es expresada en esta representación.

Fortalezas y debilidades

Fortalezas

- El artículo propone definir una estructura con conjuntos de elementos mapeables y mapeados, considerando un conjunto limitado de operaciones para expresar las transformaciones, pero que sean operaciones genéricas. Siempre se puede expresar de manera genérica y al momento de la implementación definir “módulos” específicos para casos especiales.
- Propone la opción de una metadata que exprese cierta semántica de los objetos y en un campo definir una vista que realice el mapeo específico. De cierta manera se aproxima a lo planteado por Larrañaga [Lar00] y Peralta [Per01].
- Apoya los objetivos buscados en el presente proyecto: lograr un mapeo genérico que sea independiente del origen de datos.

Debilidades

- Si bien el autor propone estructuras abstractas que permitan representar la información del mapeo, y define transformaciones a través de primitivas generales, no brinda una metodología tangible sino que sólo presenta definiciones conceptuales.

Visión general

La Figura 10 ofrece un resumen de los proyectos realizados por el grupo CSI [CSI], donde se muestra la intervención de cada uno de ellos en el desarrollo de un Datawarehouse.

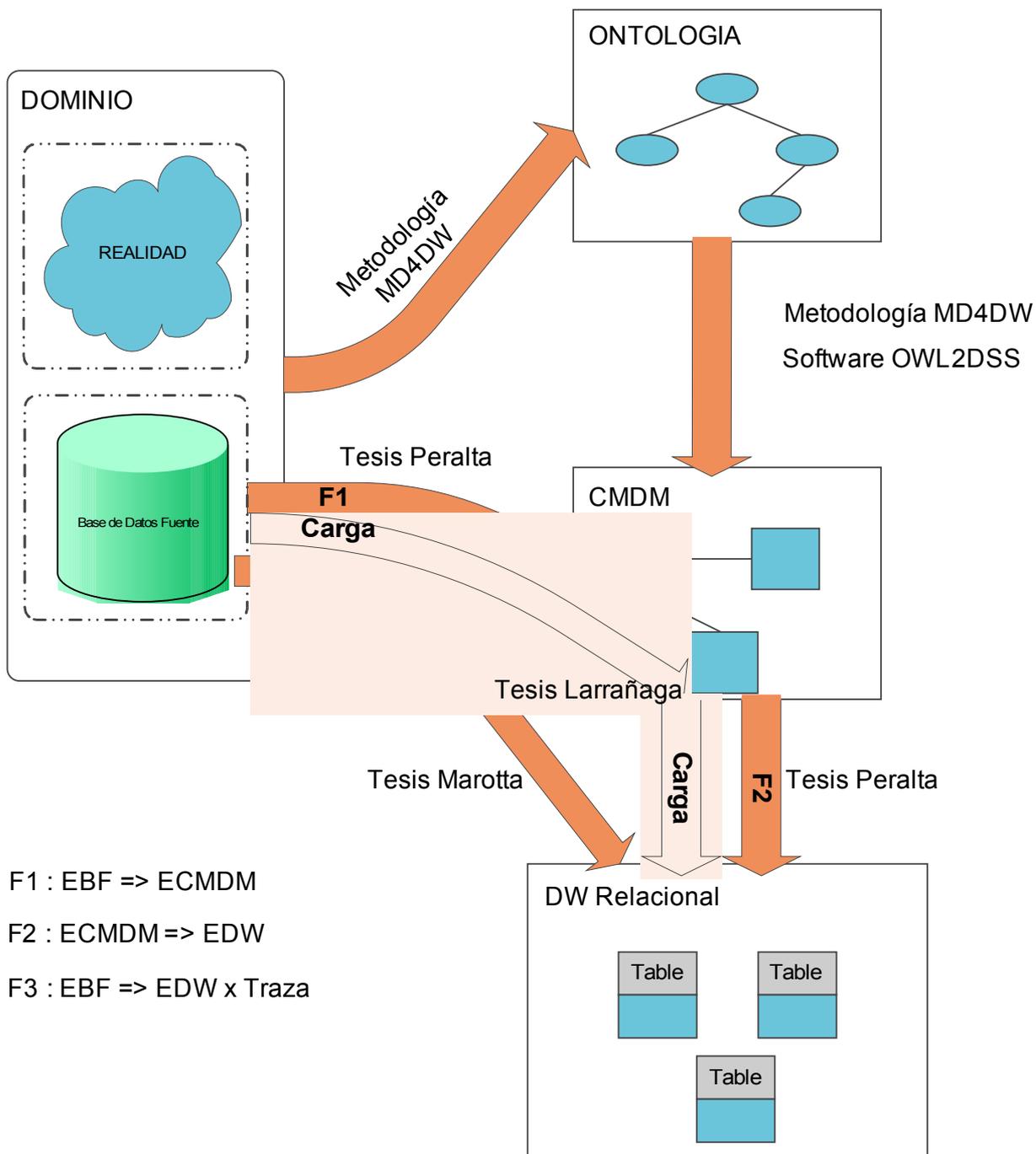


Figura 10. Trabajos de CSI relacionados con diseño de DW.

Concepción general de la solución propuesta

El objetivo de este capítulo es describir de manera general cómo fueron abordados los problemas de un proceso de construcción de Datawarehouse descritos en el capítulo “Reseña del proceso de construcción del DW”, en el intento de proponer una guía que abarque el proceso desde el relevamiento de los requerimientos hasta el diseño lógico del Datawarehouse y de su carga. Dicha guía se basa en los proyectos resumidos en el capítulo anterior, los cuales proponen herramientas para las diferentes etapas del proceso, pero en forma independiente. Este trabajo propone una nueva guía que contempla e integra las soluciones propuestas en los diferentes proyectos.

Estudios preliminares para definir el camino a seguir en este trabajo

Propuesta 1:

Inicialmente se planteó la posibilidad de extender el plugin OWL2CMDM del proyecto DSSMet [DSS06], implementando las reglas propuestas por MD4DW [MD06]. Debido a que el plugin sólo abarca el pasaje desde la ontología expresada en OWL al modelo conceptual expresado con CMDM y que MD4DW [MD06] abarca más etapas como ser el relevamiento y documentación de requerimientos, se concluyó que extender la metodología de MD4DW [MD06] representaba un aporte mayor.

Una vez descartada la alternativa de extender el plugin, se prosiguió a buscar los caminos a seguir para extender la metodología de MD4DW hasta llegar a la construcción del modelo lógico y la definición de la carga del Datawarehouse.

Propuesta 2:

Esta alternativa está compuesta por una mejora a la metodología planteada en MD4DW [MD06] y una extensión a la misma para abarcar más etapas del proceso de construcción de un DW.

Mejoras a MD4DW [MD06]

Como se mencionó anteriormente, MD4DW [MD06] abarca toda la etapa de análisis del proceso de construcción de datawarehouse, y la etapa de diseño conceptual del mismo.

El primer paso para medir las bondades que propone la metodología presentada en el proyecto, fue aplicar su propuesta en un caso de estudio (ver anexo Análisis del contexto de trabajo - Caso de Estudio).

En la Fase 1: Comprensión de la realidad, se alcanzaron los objetivos propuestos: entender y documentar la realidad bajo estudio. Esta fase se mantendría sin cambios en la solución final.

En la Fase 2: Modelado de Dominio, se detectaron las primeras carencias de la guía propuesta ya que no se logró expresar claramente en la ontología cuáles eran las claves de cada concepto que participa en la realidad. Si bien se intentó expresar dichas claves a través de una superclase Key, esta primera solución afectaba la descripción de la realidad a través de la ontología. Esto

con respecto a la validación con el cliente, dado que se agregaron conceptos que no eran directamente mapeables con objetos de la realidad.

En este trabajo se propone como solución al problema antes planteado, modificar el documento Metadata de Conceptos de la primera fase, agregando un metadato que brinde información sobre el o los atributos que identifican cada concepto. Y para representar este nuevo metadato en la ontología, se propone trabajar con OWL 2 [OWL2], que en esta nueva versión brinda primitivas para la representación de claves.

De esta manera, mediante la descripción de los conceptos en la Metadata de Conceptos y la ontología, se alcanza una descripción más completa de la realidad y sus relaciones, incluyendo las claves de cada concepto.

En la Fase 3: Modelado Conceptual Multidimensional, MD4DW propone una serie de reglas que apoyan el diseño del modelo conceptual del datawarehouse en base a la ontología y la Metadata de Conceptos. Las dos reglas que presentaron dudas fueron las referentes a la definición de dimensiones y relaciones dimensionales, reglas X y XI respectivamente. Si bien las reglas están correctamente planteadas, al momento de aplicarlas, su descripción no es lo suficientemente detallada y puede llevar a tomar decisiones de diseño inadecuadas.

Por lo tanto, la solución final de este trabajo propondrá una descripción más detallada de dichas reglas.

Extensiones a MD4DW [MD06]

Además de las alteraciones que se harían a la metodología planteada en el proyecto MD4DW [MD06], se debieron definir las extensiones que se harían a la misma.

La primera inquietud que se planteó fue con respecto al origen de datos: puede notarse que la metodología no presta atención a las fuentes de datos que poblarán el DW creando y es sabido que en un proyecto de DW gran parte del tiempo de desarrollo se gasta en integrar los orígenes de datos. Puede darse el caso de relevar cierta parte de la realidad, diseñar un modelo conceptual e incluso diseñar el DW y luego de todo este esfuerzo enterarse que los sistemas operacionales no mantienen la información necesaria para poblar el DW recién creado. Inclusive, en el caso que toda la información esté presente, los desarrolladores de la carga deben esperar a tener finalizado el diseño del DW para poder crear la ETL que lo pueble.

Dados los inconvenientes que pueden surgir de iniciar la tarea de creación de las ETL al final del proyecto, surge la pregunta: ¿No sería necesario brindar una guía o apoyo para prevenir o mitigar estos problemas?

Esta pregunta motivó el camino a seguir en este proyecto, el cual tiene dos componentes:

- Asociar el(los) origen(es) de los datos con la Metadata de Conceptos y el modelo conceptual del DW expresado en CMDM, que resultan de aplicar la metodología MD4DW.
- Alcanzar el diseño lógico de dicho Datawarehouse y el diseño de la carga del mismo en base al mapeo realizado entre el origen de datos y el modelo conceptual. De esta forma se mantendría una trazabilidad completa a lo largo del proceso y una descripción detallada del origen de datos.

La asociación entre la Metadata de Conceptos y el modelo conceptual del DW debería ser independiente del tipo de origen de datos, siendo tan genérica que permitiera trabajar con varios tipos de orígenes. Además, se hizo hincapié en la necesidad de contar con un mapeo descriptivo, que permitiera expresar en forma clara y concisa cómo se obtendría cada elemento

del modelo conceptual, expresando las transformaciones que deberían realizarse, tipo de datos, fuente, etc.

Las características anteriores son las que se encuentran en los artículos de Bernstein [Ber00] cuando éste describe la solución ideal a la generación de los mapeos que permitirían la carga de un Datawarehouse.

Para conseguir los dos objetivos descritos arriba, se plantearon los siguientes caminos:

- **Metadata y mapeos**

Generar metadatos que describan la fuente de datos y los mapeos con cada uno de los conceptos en un lenguaje que no dependa del tipo de cada fuente y guardarlos en el documento “Metadata de Conceptos”.

De esa manera, al terminar el proceso, se contaría con los datos sobre los mapeos desde las fuentes de datos y con esta información se podría realizar una ETL con posibilidad de automatizar dicho proceso.

Tal como lo plantea Bernstein en [Ber00], para soportar cualquier tipo de origen, se debería contar con un lenguaje que exprese dichas transformaciones y con algún método que permitiera implementar, para los casos más complejos, funciones de transformación específicas para cada tipo de origen.

La utilización o creación de un lenguaje con dichas características representaría un alcance más amplio que el previsto por este proyecto, por lo cual se decidió descartar este camino y seguir enfocados en los objetivos planteados anteriormente.

- **ODS**

Otro posible camino analizado luego de descartar el planteado anteriormente, fue la definición de una base de datos, más específicamente un ODS, que representara la realidad relevada en un esquema relacional. O sea, la finalidad era obtener la representación relacional de los conceptos de la ontología.

Nótese que los mapeos y transformaciones desde los distintos orígenes de datos al ODS se hacen en el lenguaje que sea más adecuado para la fuente de datos origen (por ejemplo xslt si la fuente es xml, o sql si la fuente es una base relacional, un programa si se trata de un archivo propietario, etc). Este proyecto no se enfocaría en realizar el mapeo desde los orígenes de datos al ODS, sino que el enfoque estaría en obtener un ODS que permita realizar el diseño de la carga del Datawarehouse.

Fueron estudiados algunos algoritmos para este propósito, el que más se aproximó a los objetivos de este proyecto fue el presentado en “Transforming ontology representation from OWL to relational database” [O2RD06], pero en éste no está especificado cómo se determinan las claves y por tanto no queda claro como llega a las claves foráneas. Por esto, se siguió con la investigación sobre cómo construir dicho ODS.

Dado que la ontología desarrollada aplicando la metodología presentada en MD4DW [MD06] representa la realidad sobre la cual se deben cumplir los requerimientos, el ODS debería surgir en base a la misma, por esto, se definió un algoritmo que permita generar el ODS utilizando como entrada la ontología de dominio.

De esta manera se podría proponer una solución a la construcción de un ODS que permita diseñar la carga del Datawarehouse independientemente del origen de datos.

Esta posible solución no es completa, dado que apoya la generación de la carga del Datawarehouse desde el ODS y no desde los orígenes de datos. Pero esta debilidad es mitigada por los beneficios que aporta, ya que se considera más sencillo realizar el diseño de la carga en dos etapas (“*Divide y vencerás*”), primero de los orígenes al ODS y después de éste al DW.

De esta manera los cambios que se puedan realizar sobre las bases operacionales tendrán menor impacto en el DW, pues solo deberá modificarse el mapeo al ODS.

Existe mucha literatura respecto al mapeo entre diferentes esquemas de datos, por lo que el mapeo desde los orígenes al ODS puede ser desarrollado en forma independiente y no estará dentro del alcance de este proyecto.

En cambio, sí se consideró que este proyecto debería aportar un algoritmo que permita diseñar el ODS antes descrito (ver anexo Algoritmo Onto2rdb).

Pero este enfoque aún dejaría sin solución el diseño del esquema lógico y de la carga del DataWarehouse. Luego de varios planteos, se volvieron a analizar las tesis de V. Peralta [Per01] y Larrañaga [Lar06], ya con otra visión de lo que se necesitaba.

- **ODS + Larrañaga**

El nuevo escenario con el que se cuenta ahora es un modelo conceptual del DataWarehouse expresado en CMDM y un ODS representativo del origen de datos.

A través de la propuesta de I. Larrañaga en [Lar06] se puede diseñar el modelo lógico del Datawarehouse y el diseño de la carga del mismo. Dicha propuesta brinda una versión mejorada de la presentada por V.Peralta en [Per00], tal como se describe en la sección de este documento. Por lo tanto, a través de este enfoque se brindaría los pasos necesarios para diseñar el modelo lógico de un DW, junto al diseño de la carga del mismo.

Desarrollo de la guía

Luego del estudio antes descrito, se contaba con la identificación de los proyectos sobre los cuales se iba a basar la metodología propuesta y el alcance de la misma.

Observando ahora las fases del tradicional proceso de construcción de Datawarehouse, se utilizará MD4DW [MD06] para dar apoyo a la fase de análisis y de diseño conceptual, y el algoritmo propuesto por I.Larrañaga [Lar06] para dar apoyo a la fase de diseño lógico.

A su vez, se presentará un algoritmo que permite crear un ODS a partir de la ontología desarrollada en la fase de análisis. Dicho ODS será utilizado como el origen de datos para aplicar el algoritmo de I.Larrañaga [Lar06].

Por lo tanto, este trabajo brinda una guía que cubre el proceso de desarrollo de DW basado en los requerimientos, desde la etapa de análisis hasta la etapa de diseño lógico del modelo y de su carga.

Para mostrar las bondades de la guía propuesta se aplicó la misma, esta vez completa, en el mismo caso de estudio con el que se venía trabajando.

Además de demostrar su aplicabilidad en este caso concreto, se identificaron características de la misma sobre las cuales amerita profundizar.

Dado que la metodología de MD4DW [MD06] ya había sido aplicada anteriormente sobre el caso de estudio, no se destacaron mayores comentarios al respecto, más allá de los cambios antes planteados.

Con respecto al algoritmo definido en este trabajo, Onto2rdb, que permite definir el ODS que será el origen de datos para la aplicación del algoritmo que I.Larrañaga [Lar06], se consideró que aporta valor agregado en el desarrollo del proceso de construcción del DW. Este valor agregado surge por la separación del diseño de la carga en dos etapas: el mapeo de los orígenes de datos al ODS y el diseño de la carga desde el ODS al modelo lógico del DW.

Estas dos etapas podrían realizarse en paralelo dado que una vez que se obtuvo el ODS, el mapeo de los orígenes de datos al mismo es completamente independiente del diseño de la carga del DW desde el ODS. Por lo tanto, el equipo de trabajo que se encuentre aplicando la metodología propuesta podría dividirse de manera de atacar estos problemas en caminos paralelos.

El hecho de poder dividir el equipo de trabajo en dos frentes de forma temprana tiene una ventaja muy importante: el equipo dedicado a realizar las ETL que cargan el ODS puede detectar faltantes o problemas de calidad en los datos de las fuentes de manera temprana. En caso de dejar para el final del proyecto la definición de dichas ETL, se detectarían estos problemas luego de haber terminado el modelo lógico, durante el diseño de la carga del DW. Este es un riesgo muy alto y su corrección puede tener un impacto muy grande.

Una aclaración importante es que en esta guía por razones de alcance no se atacan los problemas de calidad de datos, pero es evidente que el lugar para hacerlo sería el ODS y las ETL que lo cargan.

Continuando ahora con el algoritmo de I. Larrañaga [Lar06], a través de la aplicación del mismo en el caso de estudio se identificaron que algunos de sus pasos no son necesarios. Dado que este trabajo brinda guías para la construcción del ODS, que será el origen de datos sobre el cual se aplicará el algoritmo antes mencionado, surgen características de construcción del ODS que permiten obviar pasos propuestos por I. Larrañaga [Lar06]. En el capítulo siguiente se encuentra una descripción más amplia de los pasos antes mencionados y de la razón por la cual en este contexto ya no son necesarios.

Luego de finalizar con la aplicación de los algoritmos descritos, se tomará la Matriz de Requerimientos obtenida en la Fase 2 de la metodología MD4DW [MD06] y se extenderá la misma para incluir trazabilidad hasta los cubos obtenidos.

Esquema de la solución propuesta

Este trabajo presenta una guía para apoyar el proceso de construcción de un DW desde relevar los requerimientos, representar la realidad de la empresa, generar el diseño conceptual, el diseño lógico, hasta la carga de datos necesaria para el correcto funcionamiento del sistema.

La metodología propuesta, consta de cinco fases presentadas en la Figura 11. Error: No se encuentra la fuente de referencia:

Fase 1: Entender la realidad y documentarla en el documento Metadata de Conceptos.

Fase 2: Modelar formalmente la realidad mediante una ontología.

Fase 3: Generar el modelo conceptual expresado en CMDM.

Fase 4: Crear el ODS.

Fase 5: Generar el diseño lógico y la carga de datos.

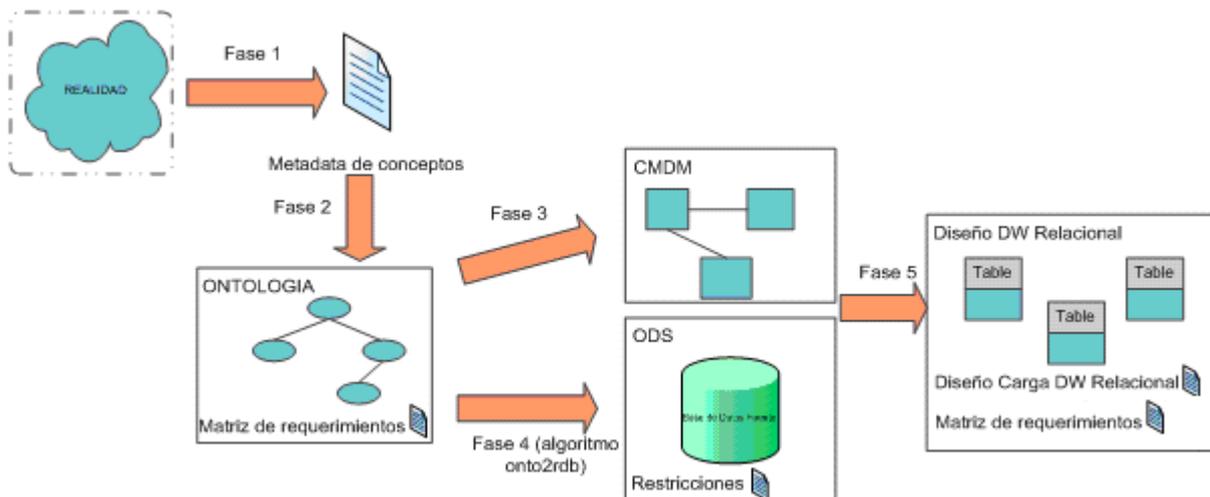


Figura 11. Fases de la guía de construcción de un DW

En cualquiera de las fases en que se esté trabajando, en caso de detectar un error, se puede volver atrás para subsanarlo, repasando las siguientes fases para revisar que los siguientes resultados sean correctos o corregirlos si no los son. Por ejemplo si en la fase 3 cuando se está construyendo el modelo conceptual se detecta que algún concepto de la realidad no fue correctamente relevado, o ha cambiado desde entonces, se retorna a la fase 1, se releva nuevamente, se corrigen los documentos Metadata de conceptos y Matriz de requerimientos, se pasa a la fase 2, se corrige la ontología y se retoma la fase 3.

Descripción detallada y análisis de la solución

El objetivo de este capítulo es justificar y explicar con ejemplos los puntos descritos en el capítulo anterior.

Para tener pleno conocimiento de la metodología presentada en MD4DW [MD06], se realizó una aplicación práctica en una empresa que tenía la necesidad de diseñar un DW. Para esto se aplicó por completo dicha metodología (ver anexo Análisis del contexto de trabajo - Caso de Estudio) y se identificaron los inconvenientes descritos en el capítulo anterior.

Primera ejecución del caso de estudio

Luego de realizar el relevamiento de requerimientos y tener listos los documentos Metadata de Conceptos y Matriz de requerimientos, se observó que las claves de los conceptos no estaban claramente definidas en ninguno de éstos documentos. En principio se dejó este punto para ser solucionado en la Fase 2 al representar la ontología, ya que este tipo de modelado permite una mejor especificación que los documentos mencionados anteriormente.

Durante la Fase 2 se encontraron dificultades para representar las claves y se tomó la decisión de que las mismas fueran representadas por conceptos y no como restricciones sobre los atributos.

En la Figura 12 se muestra la representación de las claves de los conceptos *Persona* y *Región* a través de los conceptos *Documento* y *KeyRegion* respectivamente. Los conceptos de clave son representados con una relación is-a con el concepto *Key*, del cual deben ser subclases todas las claves para dejar claramente representado que se trata de las claves de cada concepto.

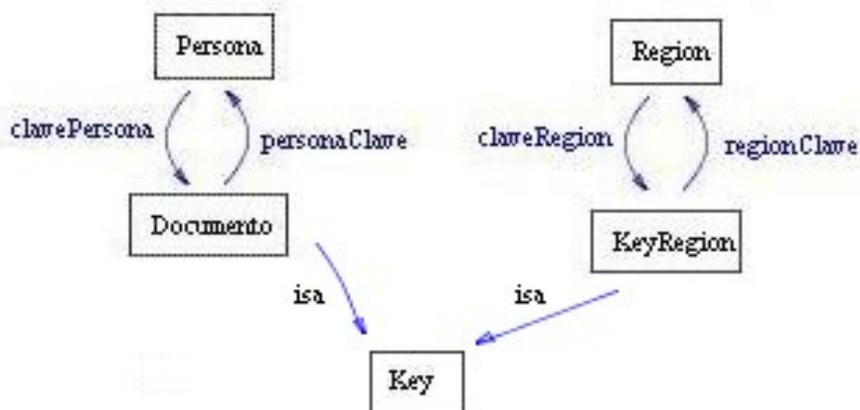


Figura 12. Representación de claves

En el caso de claves compuestas, como ser el documento de una persona, se tomó como regla incluir dichos atributos como parte del concepto que representa la clave y no como parte del concepto mismo. O sea, *Documento* tendrá dos atributos, tipo y número de documento y juntos componen la clave del concepto *Persona*.

De manera análoga se representaron las claves de todos los conceptos identificados en esta realidad.

Pero, esta solución al problema de las claves, además de convertir una ontología simple en una ontología más grande y compleja, agrega dificultad para comprenderla y muestra conceptos que en la realidad del cliente no existen como tales, sino que son restricciones sobre dichos

conceptos. Esto degrada en cierta forma la fortaleza de la ontología en cuanto a ser una gran herramienta de intercambio entre el cliente, analistas y desarrolladores.

Debido a las carencias detectadas en esta representación de las claves, se realizó un análisis de la situación y se decidió agregar un metadato en el documento de Metadata de Conceptos que permita especificar las claves de cada uno de ellos. Para esto se agregó el metadato *Identificador* como se muestra en la siguiente plantilla de Metadata de Conceptos.

Nombre:	Nombre del concepto	Fecha:	[dd/mm/aaaa]
Descripción:	<i>Descripción del concepto</i>		
Responsable:	<i>Nombre 1[, Nombre 2, ..., Nombre N]</i>		
Fuente:	<i>Origen del requisito</i>		
Usos:	<i>Cómo y donde se utiliza el concepto</i>		
Supuestos:	<i>Excepciones, observaciones, etc...</i>		
Identificador:	<i>Atributo 1[, Atributo 2, ..., Atributo N]</i>		
Atributos:			
Nombre:	<i>Nombre del atributo 1</i>		
Descripción:	<i>Descripción del atributo 1</i>		
Cardinalidad:	<i>Define la cardinalidad del concepto, es decir con cuantos elementos de este tipo se relaciona el atributo en cuestión</i>		
Dato calculado:	<i>Si corresponde a un atributo obtenido de un cálculo en función de otros conceptos, se registra la función/fórmula utilizada para calcular el mismo</i>		
Dominio:	<i>Define el conjunto de valores posibles el concepto que se define</i>		
Obligatoriedad:	<i>[Este dato es obligatorio “vacío en caso de no ser obligatorio”]</i>		
Concepto Rel. :	<i>Referencia al concepto con el que está relacionado (por ejemplo porque es foreign key)</i>		

Además de esta representación en la Metadata de Conceptos, se debió trasladar este metadato a la ontología. Para esto se realizaron cambios en la representación de la misma, eliminando los conceptos de clave y trasladando sus atributos a los conceptos de la realidad.

Para representar en la ontología los identificadores de cada concepto, se propone trabajar con OWL 2 [OWL2] y agregar restricciones mediante la primitiva *HasKey*. Representar la ontología en OWL 2 tiene como ventaja la disponibilidad de la primitiva *haskey* que permite indicar cuales propiedades identifican a un concepto y como desventaja que se encuentra menos software para trabajar. Con el correr del tiempo esperamos que haya más software para trabajar con OWL 2.

Segunda ejecución del caso de estudio

A continuación se procedió a la ejecución de la metodología MD4DW [MD06] con las alteraciones que solucionan los problemas antes descritos, para verificar la eficacia de dichas soluciones.

El relevamiento de requisitos realizado en la Fase 1 de la primera ejecución, fue reutilizado por completo, con la salvedad de que se agregó, para cada concepto, el atributo “Identificador” en la

Metadata de Conceptos. De esta forma, se tuvieron claramente identificadas las claves de todos los conceptos al momento de presentar dicho documento para validación con el cliente.

En la Fase 2, se procedió a generar la ontología sin la representación de las claves como conceptos. Como resultado, se obtuvieron las siguientes mejoras:

1. Menos clases:

Se puede apreciar que en la primera ontología, para representar la realidad se utilizaron 28 clases (si se cuenta el concepto Key, el cual fue quitado para simplificar este dibujo). De estas clases, 14 son para representar los conceptos de la realidad y las restantes 14 representan las claves de los mismos.

En cambio, la ontología sin representación de claves, sólo tiene los 14 conceptos identificados en el relevamiento de la realidad.

2. Menos relaciones:

De manera análoga, la cantidad de relaciones se reduce a menos de la mitad, ya que desaparecen las que unen cada concepto con sus claves y las existentes entre los conceptos clave y la superclase Key.

3. Todos las clases se mapean con conceptos del documento “Metadata de conceptos”:

Cada concepto que aparece en la ontología es parte de la Metadata de Conceptos, pues no son agregados nuevos conceptos para representar restricciones.

4. Simpleza y claridad:

Debido a la menor cantidad de conceptos y relaciones representados, junto con un mapeo directo entre los conceptos de la ontología y los de la Metadata de Conceptos y la Matriz de Requerimientos, esta ontología es más simple y clara para presentar al cliente para su validación.

5. Representación la ontología utilizando Protégé [Prt]:

La reducción de la cantidad de clases y relaciones, produce una reducción considerable en la creación de la ontología en Protégé [Prt].

6. Aplicación del algoritmo Onto2rdb:

Debido a la simpleza de esta nueva ontología, también se hallaron ventajas al momento de realizar la ejecución del algoritmo para generar el ODS a partir de la misma. Esta ventaja será justificada en la tercera ejecución del caso de estudio, ya que hasta esa ejecución, sólo se aplican las tres fases de MD4DW [MD06].

En la Figura 13 se pueden apreciar gráficamente las ventajas de esta nueva representación.

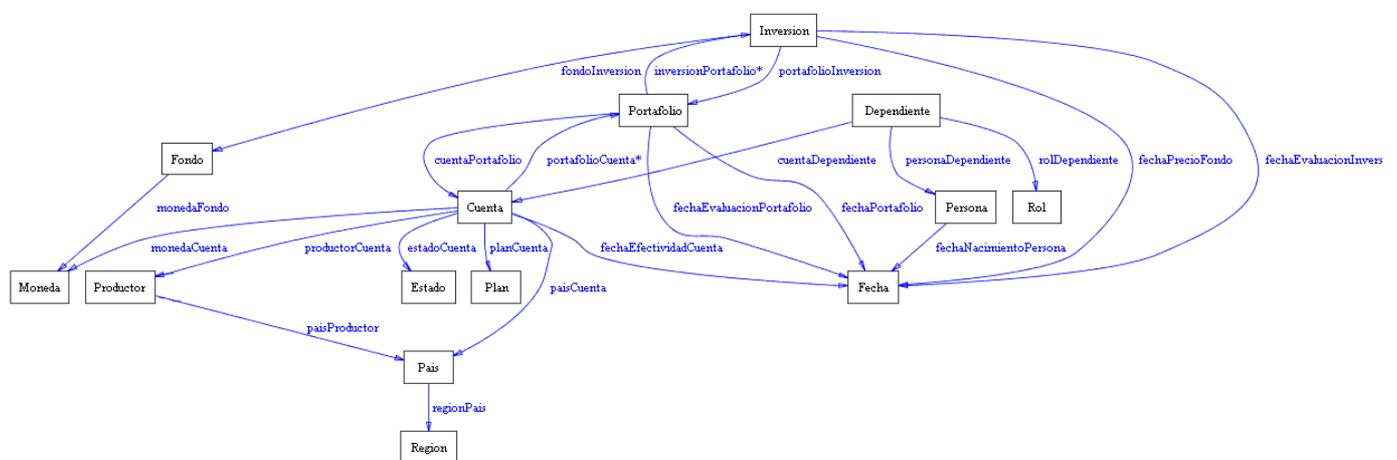
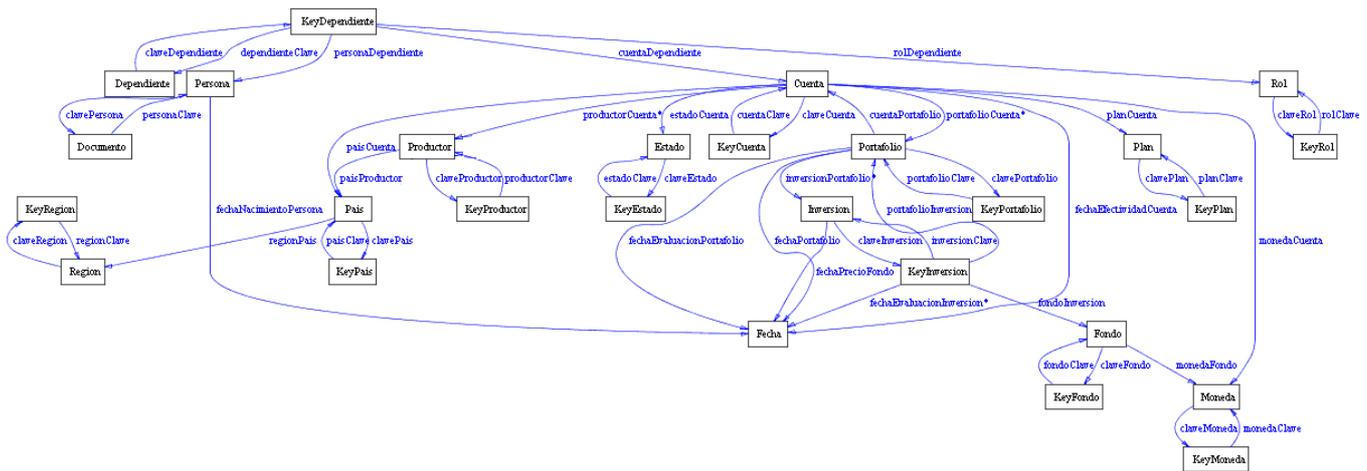


Figura 13. Comparación de ontologías

En esta ejecución se dio una solución o mejora al problema de las claves, pero se encontraron otras dificultades en la Fase 3.

Al finalizar la ejecución de dicha fase, se identificaron problemas en la aplicación de las reglas X y XI planteadas en MD4DW [MD06]. Por esto, se procedió a reescribir la Regla X y dar una explicación más detallada a la Regla XI.

El objetivo de la **Regla X** es la identificación de dimensiones del modelo multidimensional. Utilizando los niveles parcialmente ordenados, obtenidos en la aplicación de las reglas anteriores, una dimensión queda formada por el subconjunto de tuplas de niveles que tengan algún nivel en común, respetando el orden parcial previamente definido.

En este caso de estudio, debido a las relaciones existentes entre los niveles, se obtiene una única dimensión con tres jerarquías (ver Regla X en anexo Análisis del contexto de trabajo - Caso de Estudio).

Debido a las necesidades planteadas por los requerimientos, se notó que dicha dimensión no era una adecuada representación de los mismos. Por esto, se modificó la Regla X presentada en MD4DW [MD06] de manera de tomar en cuenta los requerimientos luego de obtenidas la(s) dimensión(es) resultado de dicha regla.

En la Figura 14 se muestra la Regla X modificada, donde se agrega un párrafo indicando que el diseñador, en caso de considerarlo necesario, deberá tomar decisiones sobre el resultado obtenido.

REGLA R10 – Identificación de dimensiones en CMDM
Objetivo: Identificar dimensiones del modelo multidimensional
Entrada: <ul style="list-style-type: none">• Conjunto de niveles.• Conjunto de niveles parcialmente ordenados.
Descripción: <p>Identificar las tuplas de niveles ordenados las cuales tengan algún nivel en común.</p> <p>Una dimensión queda formada por el subconjunto de tuplas de niveles previamente ordenados que tengan algún nivel en común, manteniendo el orden parcial definido entre ellos. Las dimensiones deben tener un nivel mínimo donde exista un identificador para cada elemento de la misma.</p> <p>Gráficamente la dimensión se forma uniendo las tuplas ordenadas según el nivel en común, manteniendo el orden definido.</p> <p>Todos los niveles deben formar parte de alguna dimensión. Puede ocurrir que exista una dimensión con un único nivel.</p>
<p>Si como resultado de lo anterior se obtienen dimensiones con más de una jerarquía, es tarea del diseñador, y en base a los requerimientos, decidir si lo que necesita es una dimensión con más de una jerarquía o varias dimensiones de una jerarquía.</p>
Resultado: Conjunto de dimensiones del modelo multidimensional CMDM.

Figura 14. Regla X modificada

Más adelante, en la tercera y última ejecución del caso de estudio, se aplicará esta mejora a la regla y se mostrarán los beneficios obtenidos.

Por otro lado, si bien la **Regla XI** que presenta MD4DW [MD06] no ha sido alterada, debido a que no se encontraron errores en la misma, se consideró importante realizar una breve descripción del entendimiento de la misma ya que algún concepto, como ser “Eje central”, no se encontraba detallado con claridad.

Detalle Regla XI: Esta regla guía la detección de conceptos que puedan ser tomados como punto de inicio para la definición de una relación dimensional.

Una relación dimensional en CMDM representa el conjunto de todos los cubos que se pueden construir a partir de los niveles de un conjunto de dimensiones.

La regla plantea dos escenarios: uno en el cual existe una dimensión de cuyo nivel inferior se desprenden varias jerarquías, y otro en el cual existe un conjunto de dimensiones que comparten un mismo nivel inferior.

El escenario depende directamente de las decisiones de diseño que se hayan tomado principalmente en la regla anterior. Esto es, tener una única dimensión formada por varias jerarquías cuyo nivel inferior común es el eje central, o bien tener varias dimensiones cuyo nivel inferior común es el eje central.

En cualquiera de los dos casos, el “Eje Central” será el nivel a través del cual se podrá realizar el cruzamiento de datos y será una medida del cubo.

Esta regla, aplicada al resultado de la Regla X (sin alteraciones), tiene como resultado el mostrado en la Figura 15. En este caso, se tomó como eje central el concepto *Inversión*, y se consideraron tres dimensiones: Fondo (rojo), Portafolio (azul) y Fecha (o Tiempo – verde).

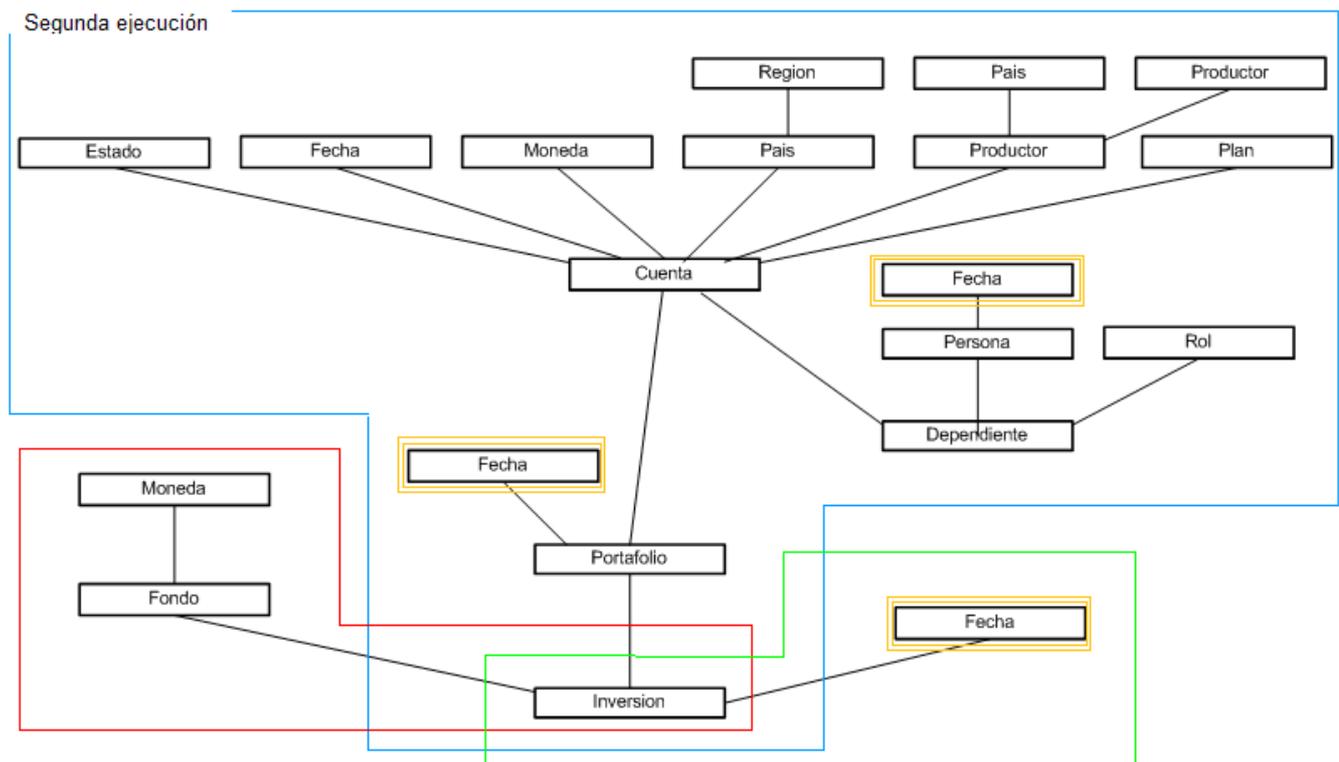


Figura 15. Resultado regla XI - Segunda ejecución

Con respecto a la dimensión **Fecha** (Tiempo de ahora en adelante), se presentó el siguiente inconveniente: se encontró el concepto *Fecha* relacionado con más de un nivel en las tres dimensiones. Esta representación múltiple del concepto *Fecha*, no permite saber cuál de ellos debe ser tomado en cuenta al momento de definir la dimensión *Tiempo*.

En la metodología MD4DW [MD06] se menciona que el concepto *Tiempo* siempre debe existir en todo DW y por esto recomienda no representar las fechas como un concepto, sino representarlas como atributos de los conceptos que corresponda. Luego, deberá ser incluido el concepto *Fecha* al obtener las dimensiones al final de la metodología.

Tercera ejecución del caso de estudio

Esta ejecución se realizó teniendo en cuenta las posibles soluciones a cada inconvenientes encontrado en las primeras tres fases. Además se ejecutan las Fases 4 y 5 y se realiza un análisis del resultado final.

Fases 1 y 2

En esta nueva ejecución de las Fases 1 y 2 se prosiguió a eliminar el concepto Fecha y la ontología resultado de éstas es la presentada en la Figura 16.

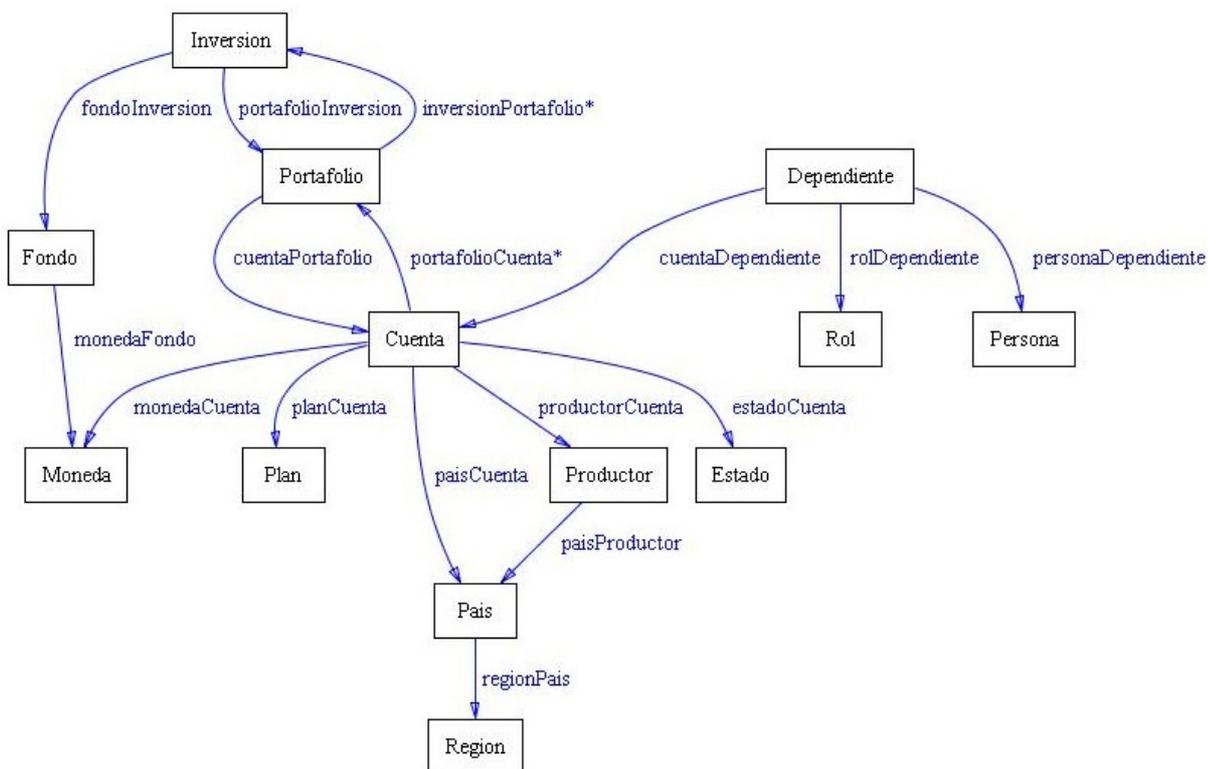


Figura 16. Ontología - Tercera ejecución

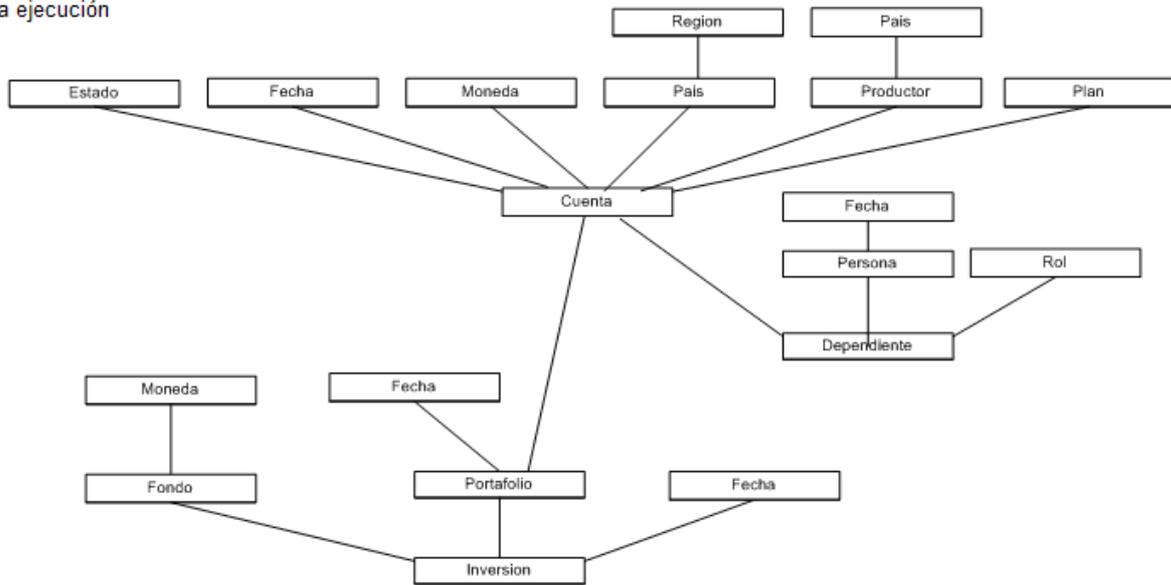
Como se puede apreciar, al quitar el concepto *Fecha*, la ontología final es aún más simple que la obtenida anteriormente y al final de la fase 3 se verificará que dicho concepto no desaparece y sí es tomado en cuenta al momento de obtener las dimensiones.

Fase 3

En esta fase se prosiguió a aplicar las reglas con las alteraciones indicadas anteriormente. A continuación, en la , se pueden apreciar las diferencias entre el resultado de la segunda y la tercera ejecución.

Las dos grandes diferencias se deben a la eliminación del concepto *Fecha* y a que se obtuvieron dos dimensiones, esto debido a que se tomó la decisión de tener varias dimensiones y no de mantener una única dimensión con varias jerarquías. Esta decisión de diseño fue tomada en base a los requerimientos planteados, por lo que se obtuvieron las dimensiones *Fondo* y *Portafolio*.

Segunda ejecución



Tercera ejecución

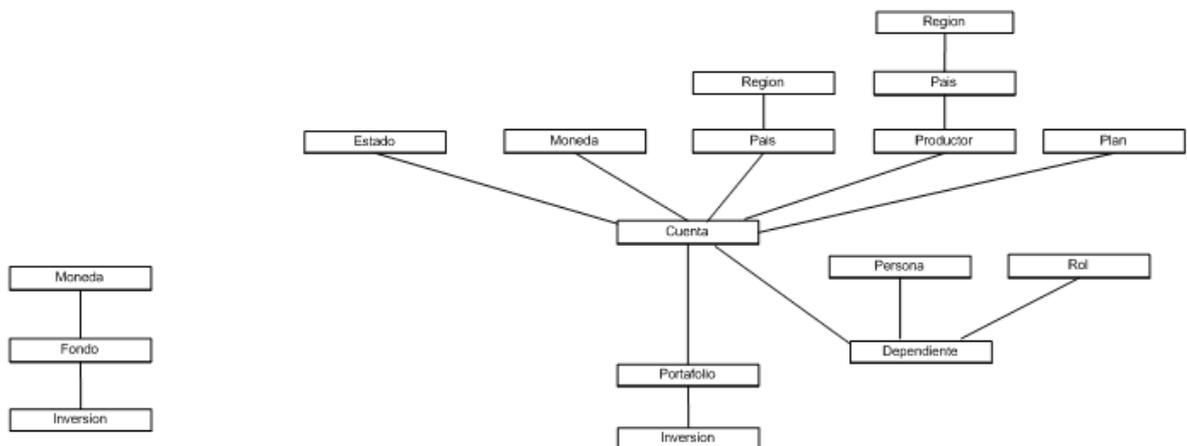


Figura 17. Regla X - Comparación de resultados

La aplicación de la Regla X, lanza como resultado dos dimensiones en el modelo multidimensional como se muestra en la Figura 18: Fondo (en rojo), Portafolio (en azul).

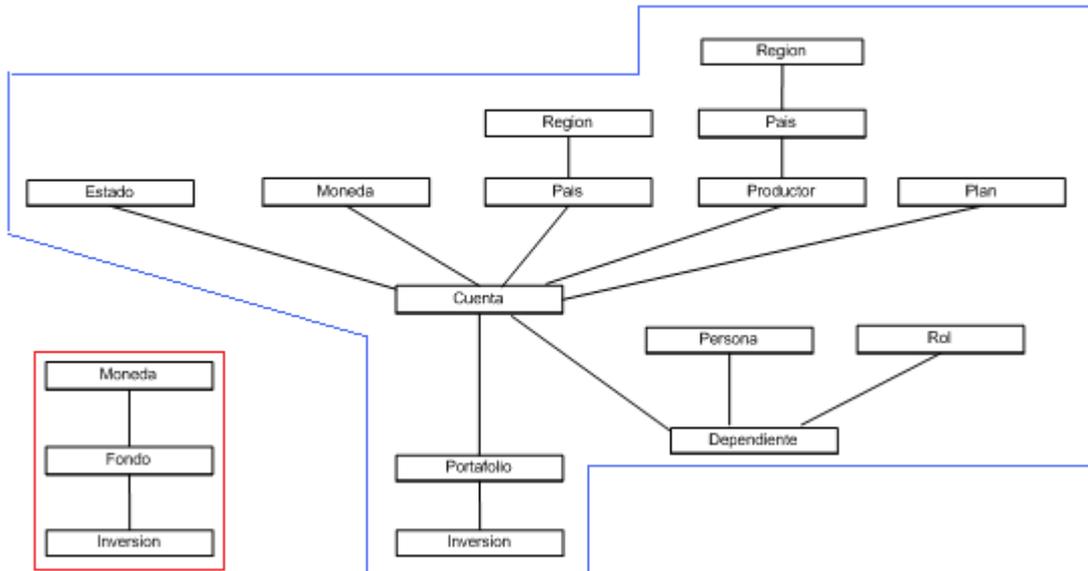


Figura 18. Regla X: Dimensiones - Tercera ejecución

Como resultado final de la aplicación de estas tres fases, se obtuvo una relación dimensional *Activos* con tres dimensiones:

1. Fondo
2. Portafolio
3. Tiempo

Las primeras dos dimensiones se han identificado teniendo en cuenta los requerimientos planteados y siguiendo la metodología planteada por MD4DW [MD06] junto a las alteraciones hechas en este trabajo.

La dimensión **Tiempo**, además de representar uno de los requerimientos, se agregó posteriormente por el supuesto de MD4DW [MD06] de que siempre estará presente en cualquier DW. Dicha dimensión está compuesta por los niveles *Inversión* y *Fecha*. Tal como se especificó en los requerimientos, la fecha sobre la cual se medirá la dimensión tiempo corresponde a la fecha de evaluación de las inversiones.

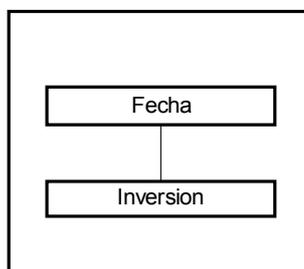


Figura 19. Dimensión Tiempo – Tercera ejecución

Relación dimensional Activos

Las tres dimensiones antes mencionadas se relacionan mediante la relación dimensional Activos como se muestra en la .

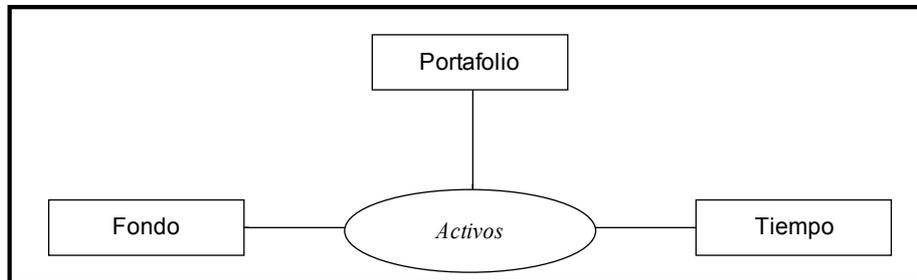


Figura 20. Tercera ejecución - Relación dimensional Activos

Como se puede apreciar, el resultado obtenido es el mismo que en las ejecuciones 1 y 2 del caso de estudio (ver anexo Análisis del contexto de trabajo - Caso de Estudio), pero con la gran ventaja de que debido a las simplificaciones encontradas, se logró una mejor representación de la realidad y una disminución en el tiempo de ejecución de las fases, debido a que se eliminaron conceptos que podían ser representados de otra manera (por ejemplo las claves y fechas).

Fase 4

Esta fase de la guía consiste en la ejecución del algoritmo **Onto2rdb**, el cual tiene como entrada la ontología definida en la Fase 2 y como salida el ODS y las restricciones necesarias. (Ver esquema de la guía en la Figura 11).

Debido a que el algoritmo se basa principalmente en la cardinalidad de las relaciones, es recomendable, antes de comenzar a ejecutarlo, listar todas las relaciones mostrando, dominio, rango y cardinalidades.

A continuación se muestran algunas filas de la tabla utilizada en la ejecución del caso de estudio (en **negrita** dos relaciones que serán tomadas como ejemplo en la siguiente página):

Relación	Dominio	Rango	Cardinalidad (Dominio)	Cardinalidad (Rango)
estadoCuenta	Cuenta	Estado	N	1
fondoInversion	Inversión	Fondo	N	1
inversionPortafolio	Portafolio	Inversión	1	N
monedaCuenta	Cuenta	Moneda	N	1
portafolioCuenta	Cuenta	Portafolio	1	N
portafolioInversion	Inversión	Portafolio	N	1
productorCuenta	Cuenta	Productor	N	1

Una vez ejecutado el algoritmo en su totalidad, se obtiene como resultado un esquema relacional, el cual es directamente mapeable con la ontología obtenida en la Fase anterior.

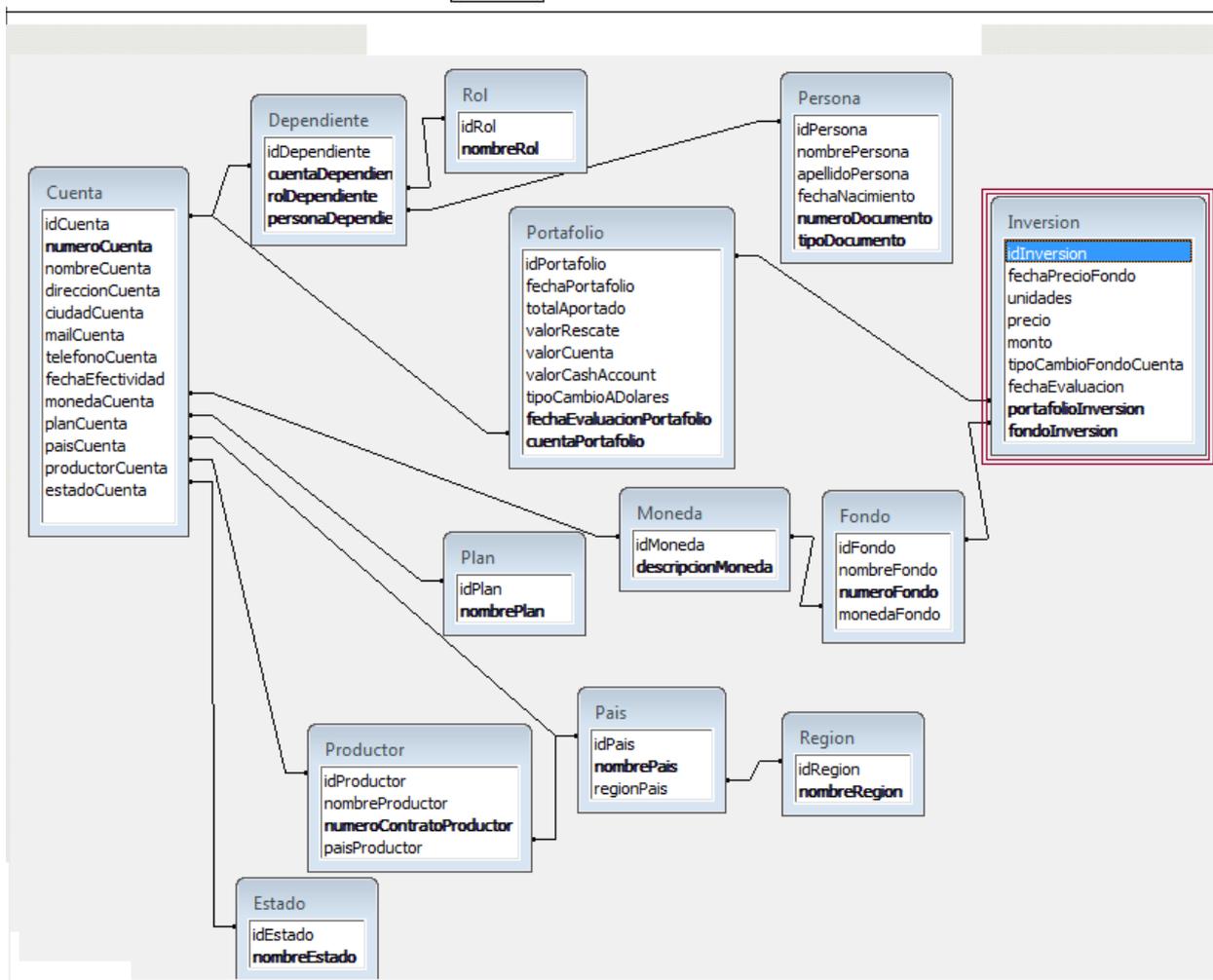
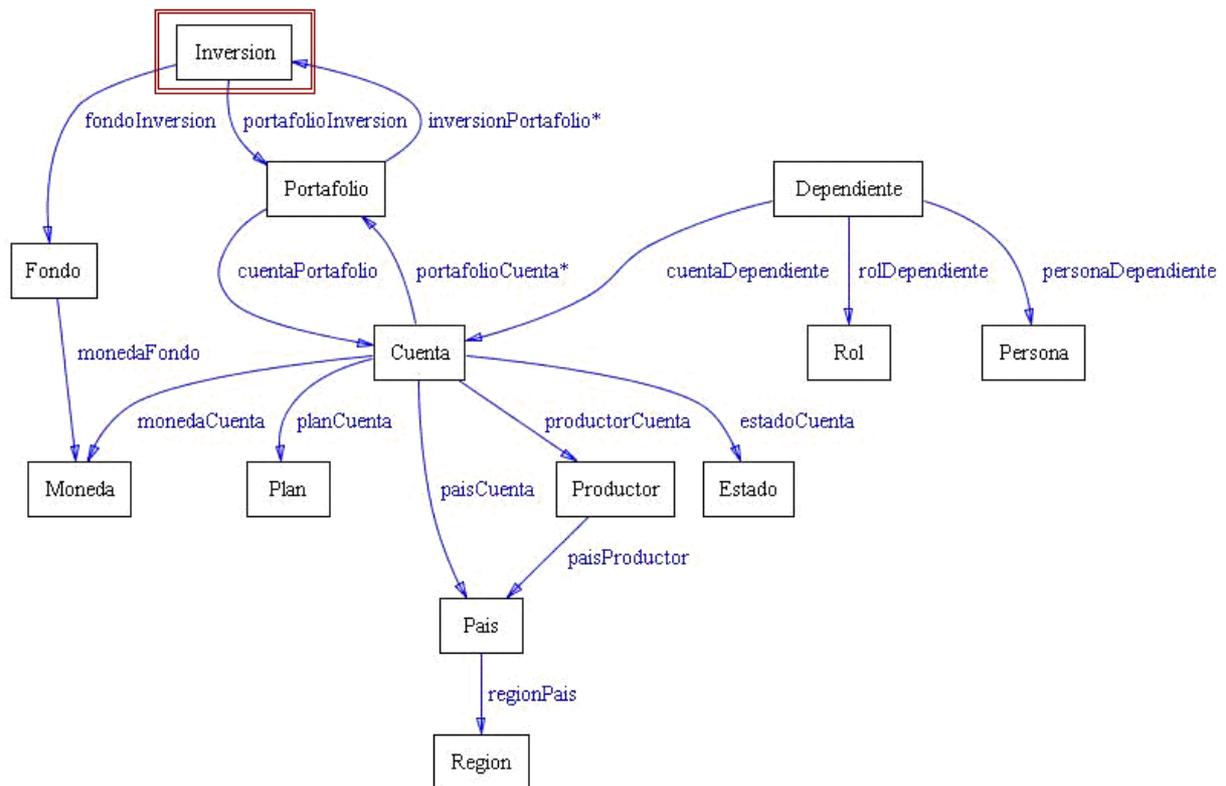


Figura 21. Comparación ODS - Ontología

En la Figura 21 se muestran la ontología y el ODS juntos, de manera de destacar la “trazabilidad” existente entre ellos. Además, se resalta en rojo el concepto *Inversión*, sobre el cual se realizarán varias observaciones.

- En la ontología, el concepto **Inversión** está relacionado con multiplicidad N:1 con los conceptos **Portafolio** y **Fondo**.
En el ODS, dichas relaciones están representadas por los atributos *portafolioInversion*, y *fondoInversion* respectivamente, los cuales son claves foráneas de las tablas Portafolio y Fondo.
- En la ontología, el concepto **Portafolio** está relacionado con multiplicidad 1:N con el concepto **Inversión** mediante la relación *inversionPortafolio*, la cual es la inversa de *portafolioInversion*.
En el ODS, dicha relación queda representada por el atributo *portafolioInversion* mencionado en el punto anterior. O sea, debido a que es la inversa, no es necesario crear un nuevo atributo para representarla.
- Por cada datatype property del concepto *Inversión* en la ontología, aparece en el ODS un atributo en la tabla con el mismo nombre.
- Todas aquellas restricciones que no fueron representadas en el ODS como tales, se encuentran en el Log de restricciones. Aquí se presenta la restricción faltante sobre la tabla de Inversiones.

Clase	Restricción	Descripción de la restricción
Inversión	$Inversion.monto = Inversion.unidades * Inversion.precio$	El monto de cada inversión es igual a las unidades de la inversión multiplicadas por el precio de las unidades de la inversión.

De manera análoga se obtuvieron todas las tablas del ODS. Ver la ejecución completa y detallada en el anexo Fase 4 - Construcción del ODS (algoritmo ONTO2RDB)

Fase 5

Avanzando en el proceso de construcción de Datawarehouse, se consideró el algoritmo propuesto por I.Larrañaga en [Lar06].

Dicho algoritmo propone partir de un modelo conceptual de Datawarehouse, representado en CMDM, y una base de datos fuente para definir un modelo lógico del Datawarehouse y el diseño de la carga del mismo.

Este algoritmo consta de cinco etapas:

- E1 - Identificación del Esquema Conceptual
- E2 - Identificación de la base de datos fuente
- E3 - Definición de Lineamientos
- E4 - Definición de correspondencias
- E5 - Aplicación del algoritmo de transformación

Etapas 1 y 2

Las dos primeras etapas se resolvieron considerando los resultados de la aplicación de MD4DW y de Onto2rdb respectivamente.

Etapa 3 - Definición de Lineamientos

Con respecto a la tercera etapa, el trabajo de I.Larranaga [Lar06] no ofrece guías para la definición de lineamientos. Esto es, no propone criterios que guíen la definición de las particiones horizontales de los cubos y de las particiones verticales de las dimensiones. Si bien existe bibliografía al respecto, se puede considerar este aspecto como un trabajo futuro valioso.

En el proyecto actual, se propone definir dichos lineamientos en base a los requerimientos y las especificaciones del cliente. Por esto se definen tres cubos (ver en anexo, E.3 Lineamientos):

- Capital Fecha
- Capital Fondo
- Capital Portafolio

En cuanto a la fragmentación de los mismos, en el cubo Capital Fondo, se define una franja para las fechas del año corriente y otra para los años anteriores como se muestra en la Figura 22. Los restantes cubos no fueron fragmentados ya que por los requerimientos no sería necesario.

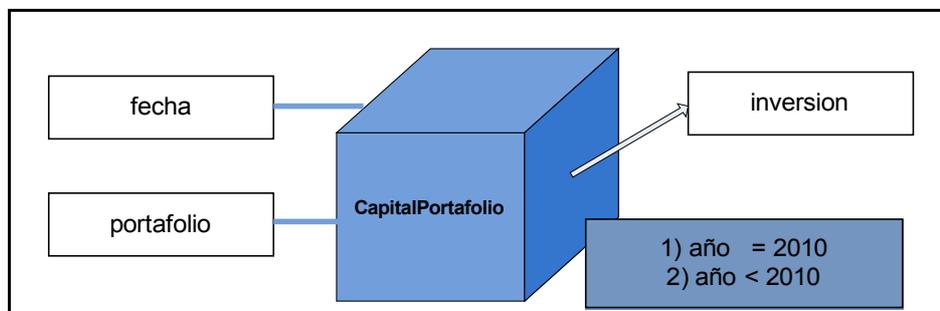


Figura 22. Cubo Capital Portafolio con Fragmentación

La dimensión portafolio ha sido fragmentada debido a la necesidad planteada en los requerimientos (ver en anexo, E.3 Lineamientos para mejor entendimiento).

E4 - Definición de correspondencias

La definición de las correspondencias entre el esquema conceptual y la base de datos fuente exige un fuerte análisis del esquema de dicha base. En un proceso clásico de construcción de Datawarehouse, luego de haber analizado la realidad del negocio, debería analizarse cómo están almacenados los datos de dicha realidad.

En la propuesta de este trabajo, dicho análisis se simplifica dado que la base de datos fuente es construida a partir de un modelo conceptual de la realidad: la ontología de dominio. De esta manera, las correspondencias entre las clases del modelo CMDM y los elementos del origen de datos pueden definirse sin necesidad de realizar el costoso trabajo de identificar cómo se almacena la información. (ver resultado de esta etapa en anexo, E.4 Correspondencias)

E5 - Aplicación del algoritmo de transformación

Las cuatro etapas antes descritas son la preparación del ambiente de trabajo para aplicar el algoritmo en esta quinta etapa.

El algoritmo propuesto agrupa sus pasos en cuatro conjuntos:

C1 - Tratamiento de dimensiones: Pasos 1 al 6

C2 - Tratamiento de cubos con mapeo simple: Pasos 7 al 12

C3 - Tratamiento de cubos con mapeo recursivo: Pasos 13 y 14
C4 - Franjas de cubos: Paso 15

En esta tercera ejecución del caso de estudio, se aplicaron los pasos definidos por I.Larañaga en [Lar06] detectando que en el tratamiento de dimensiones, los pasos 4 y 5 no son necesarios si se parte de los resultados de MD4DW y Onto2rdb.

El **Paso 4** considera la aplicación de filtros, indicando que se deben modificar las vistas para cubrir la necesidad de que las instancias deseadas para el DW cumplan ciertas restricciones, expresadas como condiciones sobre los datos de la BDF. Los filtros se agregan directamente en la vista resultado del Paso 3.

Considerando el ODS resultado de Onto2rdb como base de datos fuente, las restricciones que hayan sido relevadas fueron identificadas en el documento Metadata de Conceptos, y luego reflejadas en la ontología y descriptas en la aplicación del algoritmo Onto2rdb. Por lo tanto, toda restricción relevada de la realidad que aplicaría sobre una dimensión será implementada en el mapeo entre la base de datos de la realidad y el ODS.

El **Paso 5** considera la eliminación de los atributos que aparecen en el resultado parcial pero que no están dentro de las correspondencias definidas, es decir, que no deben estar en el Datawarehouse.

La ontología se construye en base a la Metadata de Conceptos y la Matriz de requerimientos, considerando solamente las entidades relacionadas con los requerimientos y el ODS es definido a través del algoritmo Onto2rdb a partir de la ontología antes descrita. Por lo tanto, por construcción, el ODS contiene únicamente los atributos necesarios para el DW.

Estos pasos se dejan especificados de todas formas en la solución final ya que sí deberán aplicarse en el tratamiento de cubos.

La Figura 23 presenta el esquema lógico resultado del algoritmo. En éste pueden observarse los cubos desarrollados, así como las dimensiones de los mismos y las relaciones entre ellos.

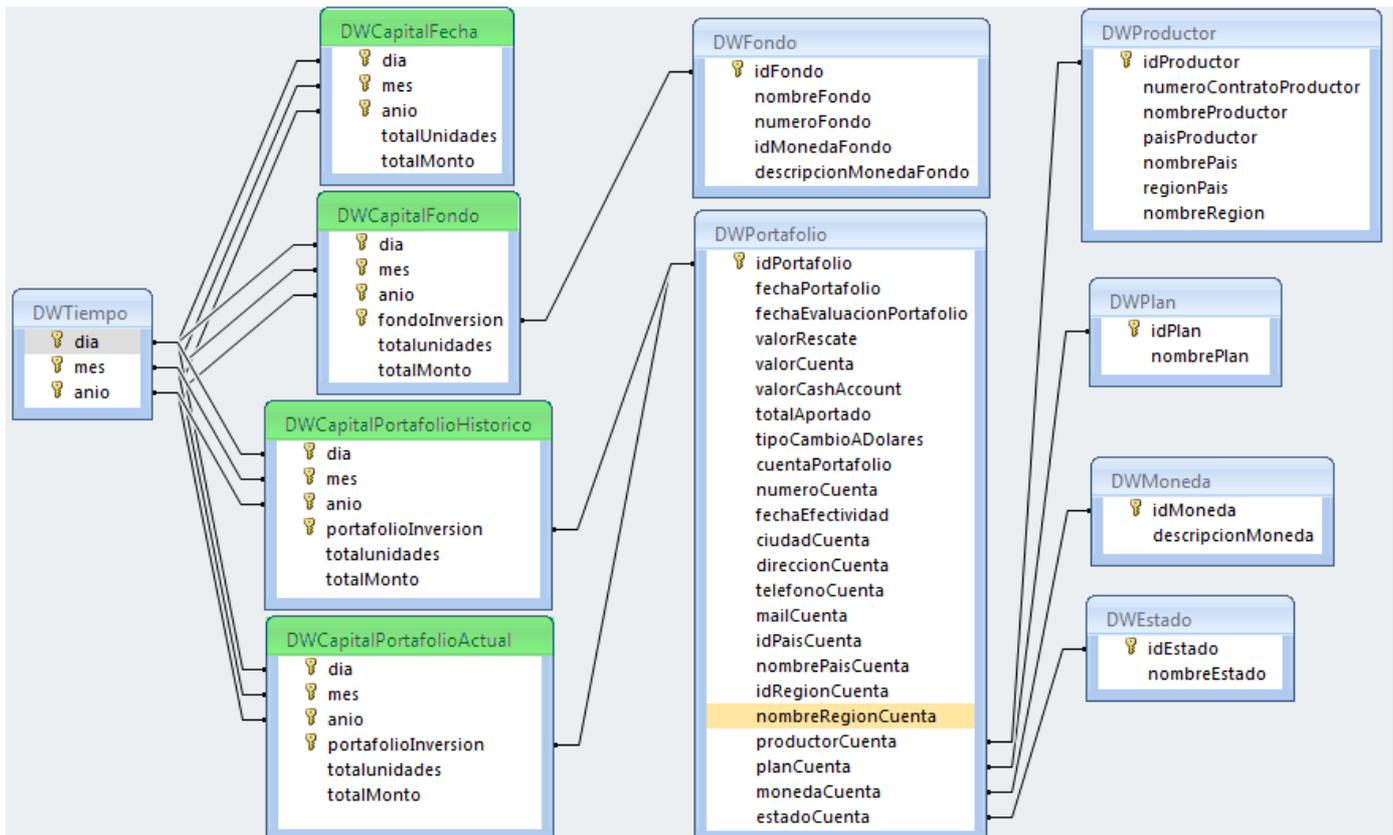


Figura 23. Tercera ejecución - Resultado Fase 5

En la siguiente tabla o matriz se muestra la trazabilidad entre los requerimientos, conceptos y cubos de esta realidad. Esta matriz tiene dos posibles lecturas, según la alteración que se desee estudiar. A continuación se presentan dos ejemplos:

- Si el concepto *Cuenta* cambia debido a una alteración en la realidad, es posible medir el impacto de dicho cambio sobre los cubos definidos en la columna correspondiente (marcados con negrita).
- Si el requerimiento RQ1 cambia, se puede realizar un filtro por la columna correspondiente y además de saber los conceptos relacionados, se pueden ver los Cubos que podrían sufrir alteraciones.

Concepto	RQ1	RQ2	Cubos
Cuenta		√	DWCapitalPortafolioActual DWCapitalPortafolioHistorico DWCapitalFondo DWCapitalFecha
Dependiente		√	
Estado		√	DWCapitalPortafolioActual DWCapitalPortafolioHistorico DWCapitalFondo DWCapitalFecha

Fondo	√	√	DWCapitalPortafolioActual DWCapitalPortafolioHistorico DWCapitalFondo DWCapitalFecha
Inversión	√	√	DWCapitalPortafolioActual DWCapitalPortafolioHistorico DWCapitalFondo DWCapitalFecha
Moneda	√	√	DWCapitalPortafolioActual DWCapitalPortafolioHistorico DWCapitalFondo DWCapitalFecha
País		√	DWCapitalPortafolioActual DWCapitalPortafolioHistorico DWCapitalFondo DWCapitalFecha
Persona		√	
Plan		√	DWCapitalPortafolioActual DWCapitalPortafolioHistorico DWCapitalFondo DWCapitalFecha
Portafolio	√	√	DWCapitalPortafolioActual DWCapitalPortafolioHistorico DWCapitalFondo DWCapitalFecha
Productor		√	DWCapitalPortafolioActual DWCapitalPortafolioHistorico DWCapitalFondo DWCapitalFecha
Región		√	DWCapitalPortafolioActual DWCapitalPortafolioHistorico DWCapitalFondo DWCapitalFecha
Rol		√	

Esta matriz, se construye a partir de la Matriz de Requerimientos resultado de la Fase 2. Sólo es necesario tener los cubos ya definidos y agregar una columna para que se pueda relacionar cada concepto con el(los) cubo(s) en que participa.

Conclusiones

Se consiguió definir una guía que mediante cinco fases y a partir de los requerimientos genera el modelo conceptual, el modelo lógico y la carga parcial de un datawarehouse. Se mantiene la trazabilidad, tanto entre requerimientos y conceptos, como entre requerimientos y cubos para poder localizar los efectos de un cambio en los requerimientos o en la realidad.

Se menciona la carga como parcial porque como ya se mencionó antes, solo se define la misma desde el ODS al datawarehouse, dejando de lado la carga desde los orígenes de datos al ODS. Si bien no se soluciona totalmente la carga del DW se consigue parte de este objetivo y además se puede comenzar tempranamente en el proyecto la parte que no está solucionada, lo cual es de gran valor para detectar problemas tempranamente con las fuentes de datos.

Como principales fortalezas de esta guía se encuentra:

- La guía se basa en los requerimientos y no en las fuentes como algunas otras guías o metodologías, lo cual promueve un enfoque en lo que los usuarios necesitan.
- Se documenta la realidad de manera que se pueda validar el entendimiento de la misma y los requerimientos desde un principio. De esta manera se minimiza el riesgo de retrocesos durante el desarrollo del proyecto, debidos a una mala comprensión o ambigüedades durante el relevamiento de los conceptos del dominio.
- La documentación generada promueve una gestión del conocimiento beneficiosa para toda la organización, y en particular disminuye el impacto de la integración de nuevos integrantes en el equipo de desarrollo.
- La trazabilidad definida a lo largo de las fases propuestas, y la documentación generada en cada una de ellas, facilita el mantenimiento del DW y la identificación del impacto de cambios que puedan solicitarse sobre el mismo.
- La división de la resolución de la carga en dos partes (de las fuentes al ODS y de éste al DW) permite que puedan ser atacados por equipos diferentes y de forma temprana.
- El diseño del modelo lógico del DW es independiente de las fuentes de datos.
- Deja planteado el problema de la calidad de datos y el lugar para resolver el mismo: el ODS y las ETL que cargan el mismo.
- Se aplicó la guía propuesta a un caso práctico real con resultados satisfactorios.
- Se definió el algoritmo onto2rdb el cual es automatizable, recibe una ontología y genera el ODS que almacena instancias de los conceptos. Este algoritmo fue probado con casos reales con resultados satisfactorios.

Trabajos futuros

Quedan planteados algunos trabajos para continuar trabajando en el futuro:

- Calidad de datos. Este suele ser un problema complejo en un sistema de este tipo y en esta metodología está claro donde debe atacarse el problema.
- Carga del ODS. Si se agregara a la presente guía la carga del ODS, se tendría una guía que contempla la carga completa del DW.
- Análisis de impacto de un cambio. Sería bueno hacer una prueba del impacto de cambios en la realidad y en los requerimientos.
- Restricciones. Muchas de las restricciones que pueden existir en la realidad pueden representarse en la ontología y también pueden representarse en CMDM, pero no en todos los casos las restricciones de la ontología son “propagadas” al CMDM y por tanto al DW, sería bueno estudiar qué restricciones pueden respetarse.
- Automatizar onto2rdb. Sería bueno automatizar la ejecución de este algoritmo para ahorrar trabajo y agilizar las partes en las que no hay necesidad de tomar decisiones, como lo es este algoritmo.

Glosario

Datawarehouse (Almacén de datos):

Colección de datos orientada a dar apoyo a la toma de decisiones en determinado contexto (empresa, organización, etc...). No muestra sólo una foto de los datos de la organización en determinado momento, sino que permite ver el histórico y evolución de los mismos.

ODS (Operational Data Store - Almacén operacional de los datos)

Contenedor de datos operacionales que ayudan al soporte de decisiones y a la operación. Su función es integrar los datos provenientes de numerosas fuentes de datos, pero con una ventana de actualización (apenas algunos minutos) y detalles mucho menores que un DW.

Fuente de datos

Entidad o sistema de archivos electrónico del cual se pueden obtener datos. Pueden ser, bases de datos relacionales, xml, archivos de texto, planillas Excel, etc...

Ontología

Especificación explícita y formal de una conceptualización, la cual tiene la finalidad de facilitar la comunicación y el intercambio de información entre diferentes sistemas y entidades.

CMDM (Conceptual MultiDimensional Model)

Modelo conceptual en donde es posible especificar una determinada parte de la realidad, con un alto nivel de abstracción y en términos multidimensionales [Car00].

Metadata (metadatos)

Información estructurada, la cual puede ser procesada automáticamente y usada para dar soporte a distintas operaciones. De una manera más informal, se puede entender el concepto metadata como información de contexto, descriptiva y administrativa sobre los datos.

Matriz de requerimientos

Refleja la relación entre los conceptos y relaciones del modelo de dominio con los requerimientos relevados.

Proceso

Un Proceso es un conjunto de decisiones, actividades y tareas que se encadenan en forma secuencial y ordenada para conseguir un resultado que satisfaga ciertas necesidades.

Proceso de carga del DW

Proceso de extracción los datos desde donde residen (fuente de datos) y transformación de los mismos para almacenarlos en el DW.

Proceso de actualización del DW

Proceso utilizado para mantener los datos actualizados en el DW, muy importante debido al constante cambio e incremento de datos en las distintas fuentes de datos.

Herramientas OLAP (On-Line Analytical Processing / Tratamiento Analítico En Línea)

Proporcionan una de las más importantes funciones de análisis para los niveles de generación de información y soporte de decisión, que es la posibilidad del análisis multidimensional. Las aplicaciones OLAP son la principal herramienta de los sistemas de soporte de decisión (DSS)

DSS (Decision Support System)

Conjunto de programas y herramientas que permiten obtener de manera oportuna información que se requiere durante el proceso de toma de decisiones que se desarrolla en un ambiente de incertidumbre [Coh99].

Anexos

Análisis del contexto de trabajo - Caso de Estudio

Aplicación de la metodología MD4DW

El caso de estudio es aplicado en la empresa uruguaya AIVA [AIVA], la cual se centraliza en carteras de inversión en todo Latinoamérica. Se le planteó al departamento de IT la posibilidad de realizar un caso de estudio en dicha empresa, y luego de analizar distintas opciones fueron seleccionados los requerimientos descritos a continuación tal cual fueron redactados por el cliente.

“El departamento de Asset Management de AIVA necesita saber cuánto dinero se encuentra invertido en cada fondo de inversión, y cuáles son las cuentas que tienen inversión en dichos fondos. Para cada fondo se debe conocer su número, su nombre y su moneda, y de cada cuenta sus productores, su número, su moneda, su plan, su datos de contacto, los titulares, su fecha de efectividad, las unidades invertidas en el fondo, el valor en dólares de la inversión en el fondo, la región del productor, la fecha de evaluación de las unidades, el tipo de cambio, y el valor en dólares de la inversión total de la cuenta. Para las inversiones en cash, debe aparecer la suma de lo invertido (en la moneda de la cuenta) bajo el fondo “Cash Account” y no se consideran unidades.”

Primera ejecución del Caso de Estudio

Fase 1 - Entender y documentar la realidad

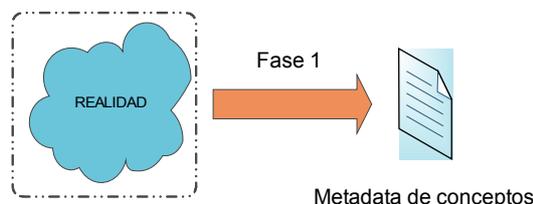


Figura 24. MD4DW - Fase 1

Como se muestra en la Figura 24, en base a la realidad planteada por parte del cliente, se realizó un análisis del dominio. Dado que la realidad del negocio es sumamente extensa, se redujo la misma para considerar los conceptos relacionados con los requerimientos descritos.

A continuación, se describen de manera más detallada la realidad planteada y los requerimientos sobre la misma.

Descripción de la realidad

En la realidad planteada existe un conjunto de fondos de inversión. Cada fondo es identificado por un número e interesa conocer su nombre y su moneda. Existen cuentas que invierten en dichos fondos. Cada cuenta se identifica por un número e interesa conocer su moneda, su plan, su estado, sus datos de contacto (compuestos por dirección, ciudad, y país, mail y teléfono) y su fecha de efectividad. Una cuenta tiene asociada un conjunto de dependientes de la misma.

Cada dependiente es una persona con un determinado rol en la cuenta. Los roles pueden ser Titular o Beneficiario. De cada persona interesa conocer su nombre, su apellido y su fecha de nacimiento.

Una cuenta es vendida por un productor. De cada productor interesa saber su nombre, su número de contrato con AIVA, su país y su región. La región del productor es la región a la que pertenece el país de este productor.

Diariamente se recibe y almacena información de las inversiones que forman el portafolio de cada cuenta. Dicha información indica cuántas unidades tiene cada cuenta en cada fondo de inversión, el precio de cada unidad, la fecha del precio de cada unidad, el tipo de cambio para pasar de la moneda del fondo a la moneda de la cuenta, y el valor total de la inversión en el fondo (igual a unidades por precio). Además se indican los valores totales del portafolio de la cuenta (en la moneda de la misma) que son el total aportado de la cuenta, el valor total de la cuenta, el valor de rescate de la misma, el valor total invertido en cuentas corrientes o plazo fijo ("Cash Accounts"), el tipo de cambio para pasar de la moneda de la cuenta a dólares y la fecha a la cual fueron evaluados dichos totales.

Dado el volumen de información con el que se cuenta, cabe agregar que la mayoría de los análisis (estimado en un 80% de los mismos) que incluyen información a nivel de cuenta se realizan sobre evaluaciones del año corriente.

Los análisis de información se realizan sobre una fecha en particular de evaluación de inversiones.

Requerimiento 1

Medir el capital por fondo: para cada fondo, la suma de todas las inversiones realizadas en el mismo a una determinada fecha. Debe considerarse como un fondo las inversiones en "Cash Accounts". Cash Account es un concepto de inversión dentro del portafolio de una cuenta (ver mayor detalle en el documento de Metadata de conceptos).

Información a desplegar:

- Número de fondo
- Nombre del fondo
- Moneda del fondo
- Total invertido en el fondo, en la moneda del fondo
- Total invertido en el fondo, en dólares
- Fecha de evaluación del fondo
- Fecha de la inversión
- Tipo de cambio de la moneda del fondo a dólares

Requerimiento 2

Conocer las cuentas que invierten en cada fondo: Para cada fondo, se debe poder llegar al detalle de las cuentas que tienen una inversión en dicho fondo, a una determinada fecha.

Información a desplegar

- Número de fondo
- Nombre del fondo
- Moneda del fondo
- Total invertido en el fondo, en la moneda del fondo
- Total invertido en el fondo, en dólares
- Fecha de evaluación del fondo
- Fecha de la inversión
- Unidades en el fondo
- Precio de cada unidad del fondo
- Productor de venta de la cuenta
- Número de cuenta
- Nombre de la cuenta
- Plan de la cuenta
- Estado de la cuenta
- Moneda de la cuenta
- Fecha de efectividad de la cuenta
- Dirección de la cuenta
- País de la cuenta
- Teléfono de la cuenta
- Mail de la cuenta
- País del productor de la cuenta
- Región del país del productor de la cuenta
- Valor de cuenta de la cuenta
- Tipo de cambio de la moneda del fondo a dólares

En esta fase queda disponible el documento de Metadata de Conceptos (Detalle en Anexo Metadata de conceptos).

Fase 2 - Modelado del dominio

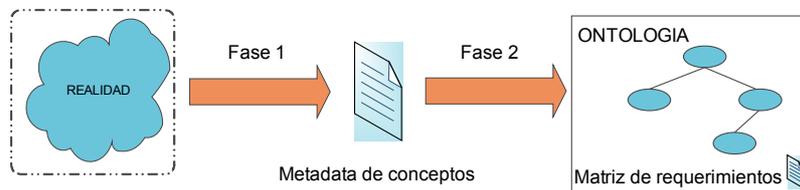


Figura 25. MD4DW - Fase 2

Como se indica en la Figura 25, una vez comprendida la realidad dentro de la cual surgen los requerimientos antes descritos, se prosiguió con la creación de un modelo formal que represente dicha realidad. Dicha representación se realizó mediante una ontología como lo indica la metodología MD4DW [MD06].

El primer problema a resolver fue el de representar de forma satisfactoria y comprensible las claves y fechas que se desprenden de la realidad.

Con respecto a la representación de claves, para cada clase dentro de la ontología se definió un nuevo concepto que represente la clave de dicho concepto, siendo todas las clases representantes de claves subclases de una clase general Key. Con esta solución, se resolvió la representación de claves simples y múltiples dentro de la ontología.

Para las distintas fechas que aparecen dentro de la realidad relevada, se decidió que serían representadas mediante un concepto *Fecha*, y no como atributos de clases. Esta decisión surge de la importancia del concepto dentro de la realidad, ya que varios conceptos progresan a lo largo del tiempo (existe una instancia por cada fecha) y desde un principio se puede inducir que se contará con una dimensión Tiempo en el modelo multidimensional final.

En la Figura 26 se muestra la representación de la ontología obtenida al realizar un análisis exhaustivo de los requerimientos antes descritos.

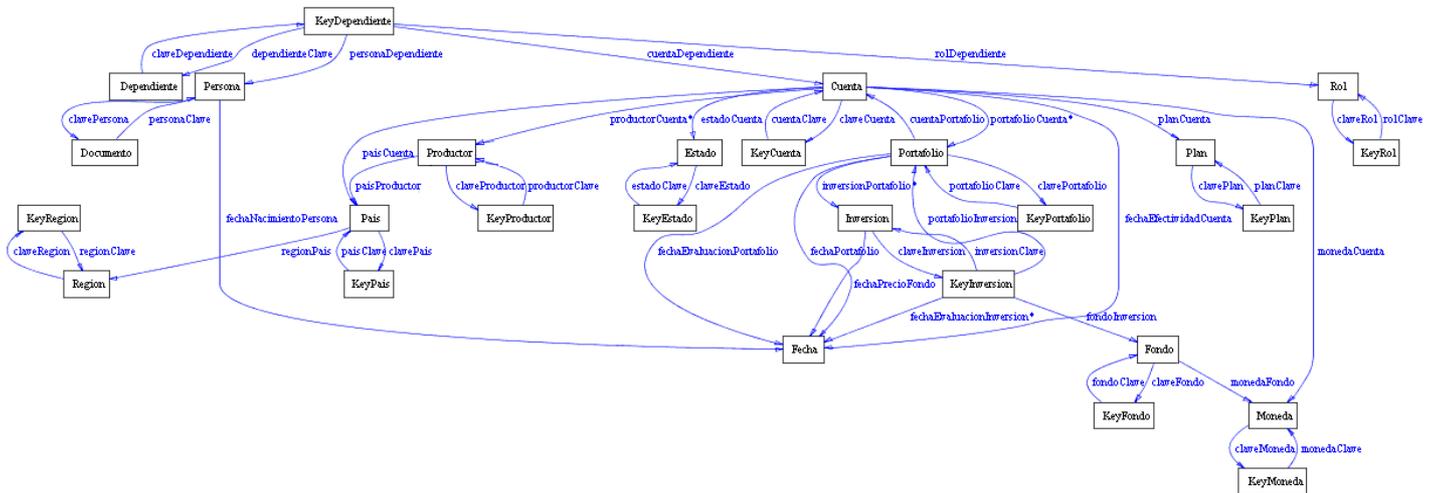


Figura 26. MD4DW - Ontología

Para la representación gráfica de la ontología se utilizó Protégé 3.4 [Prt] y la Figura 26 es obtenida en el Tab “Ontoviz”.

De manera de simplificar el dibujo se excluyó del mismo el concepto *Key* antes mencionado, pero todos los conceptos *KeyXXXX* tienen una relación con el concepto *Key* aunque aquí no aparezca representado.

En la Figura 27 se muestra la representación de la clave del concepto *Región* como un concepto *KeyRegión* el cual se relaciona con *Región* mediante relaciones funcionales, esto permite representar que cada concepto está relacionado de forma única con su concepto clave. De la misma manera se representaron las claves de todos los conceptos de la ontología.

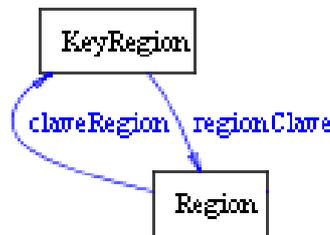


Figura 27. Representación de las claves

En esta fase, además de la ontología se genera la siguiente matriz de requerimientos:

Concepto	RQ1	RQ2
Clases		
Cuenta		√
Dependiente		√
Documento		√
Estado		√
Fecha	√	√

Fondo	√	√
Inversión	√	√
Moneda	√	√
País		√
Persona		√
Plan		√
Portafolio	√	√
Productor		√
Región		√
Rol		√
Relaciones		
cuentaPortafolio		√
dependienteCuenta		√
estadoCuenta		√
fechaEfectividadCuenta		√
fechaEvaluacionInversion	√	√
fechaEvaluacionPortafolio	√	√
fechaNacimientoPersona		√
fechaPortafolio	√	√
fechaPrecioFondo	√	√
fondoInversion	√	√
inversionPortafolio	√	√
monedaCuenta		√
monedaFondo	√	√
paisCuenta		√
paisProductor		√
personaDependiente		√
personaDocumento		√
planCuenta		√
portafolioCuenta		√
portafolioInversion	√	√
productorCuenta		√
regionPais		√
rolDependiente		√

Fase 3 - Construcción del modelo conceptual

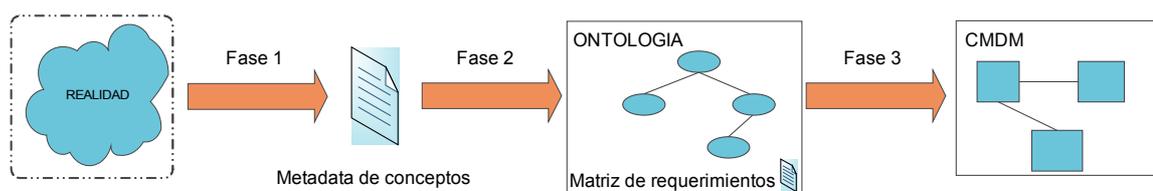


Figura 28. MD4DW - Fase 3

Según se muestra en la Figura 28, la metodología MD4DW [MD06] indica que una vez obtenido el documento de Metadata de Conceptos, la matriz de requerimientos y la ontología del dominio, se debe proceder a aplicar las reglas definidas en dicha metodología.

El resultado esperado luego de la aplicación de las reglas es el modelo conceptual del datawarehouse correspondiente a los requerimientos del caso de estudio.

Regla I: Clases a mapearse como niveles en CMDM

Se determinaron las clases identificadas en la matriz de trazabilidad para los dos requerimientos del caso de estudio. Todas estas clases mantienen una relación con su clase *Key* correspondiente, por lo cual todas las subclases de *key* se agregan al conjunto de clases candidatas a niveles. Además, se agrega a dicho conjunto la superclase *Key*, ya que representa una categorización de las claves.

Una vez aplicada esta primera regla sobre el caso de estudio, se obtuvo el siguiente conjunto de clases candidatas a niveles:

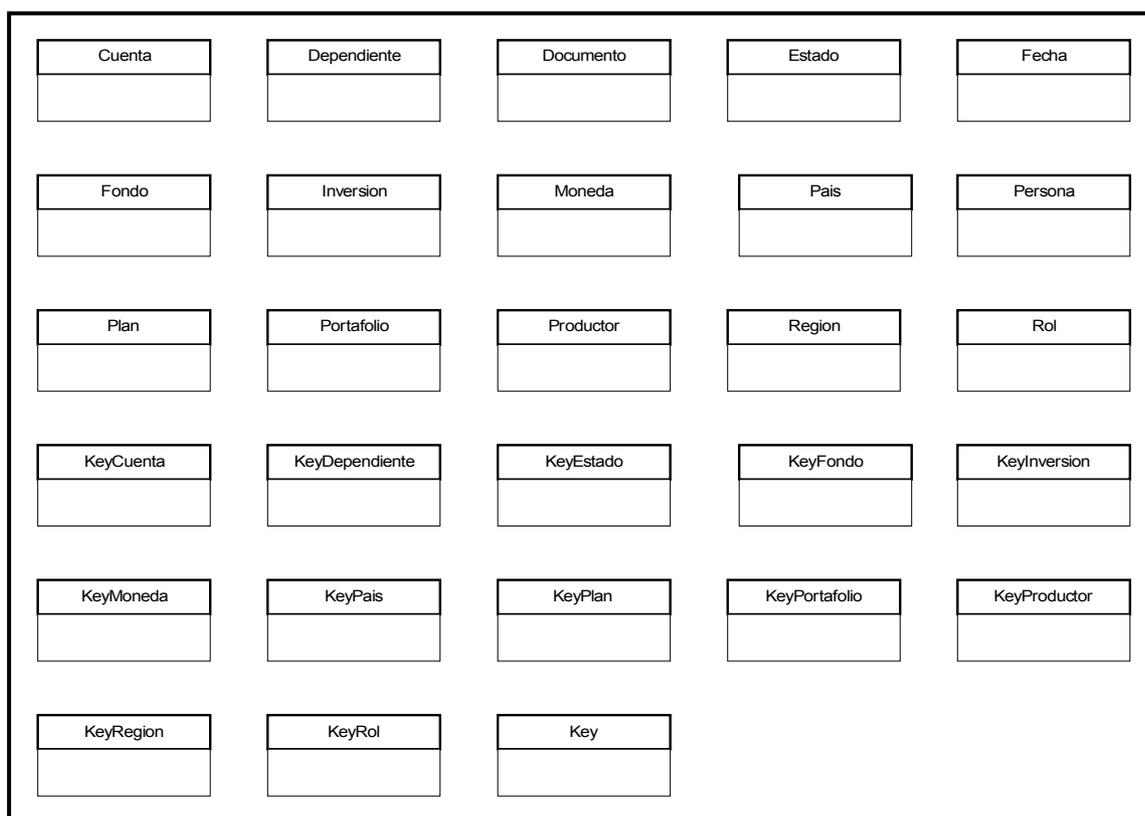


Figura 29. Regla I: Clases candidatas a niveles

Regla II: Definición de niveles en CMDM

El resultado de la aplicación de esta regla define el conjunto de niveles indicando para cada uno su nombre y sus atributos simples.



Figura 30. Regla II: Conjunto de niveles con atributos

Regla III: Definición de cardinalidades

Como lo plantea la regla, se identificaron todas las relaciones entre clases candidatas a niveles, y se definió para cada relación su cardinalidad. Resultado de la aplicación de esta tercera regla:

Relación	Dominio	Rango
claveCuenta	1	1
estadoCuenta	N	1
fechaEfectividadCuenta	N	1
monedaCuenta	N	1
paisCuenta	N	1
planCuenta	N	1
portafolioCuenta	1	N
productorCuenta	N	1
claveDependiente	1	1
claveEstado	1	1
claveFondo	1	1
monedaFondo	N	1
claveInversion	1	1
fechaPrecioFondo	N	1
claveMoneda	1	1
clavePais	1	1
regionPais	N	1
clavePersona	1	1
fechaNacimientoPersona	N	1
clavePlan	1	1
clavePortafolio	1	1
cuentaPortafolio	N	1
fechaEvaluacionPortafolio	N	1
fechaPortafolio	N	1
inversionPortafolio	N	1
claveProductor	1	1
paisProductor	N	1
claveRegion	1	1
claveRol	1	1
personaClave	1	1
cuentaDependiente	N	1
dependienteClave	1	1
personaDependiente	N	1
rolDependiente	N	1
estadoClave	1	1
fondoClave	1	1
fechaEvaluacionInversion	N	1
fondoInversion	N	1
portafolioInversion	N	1
inversionClave	1	1
monedaClave	1	1
paisClave	1	1
planClave	1	1
portafolioClave	1	1
productorClave	1	1
regionClave	1	1
rolClave	1	1

cuentaClave	1	1
-------------	---	---

Regla IV: Identificación de orden entre niveles a partir de cardinalidad N:1

Una vez identificadas las relaciones con cardinalidad N:1 de la *Regla III*, se establece un primer orden entre los niveles definidos en la *Regla II*:

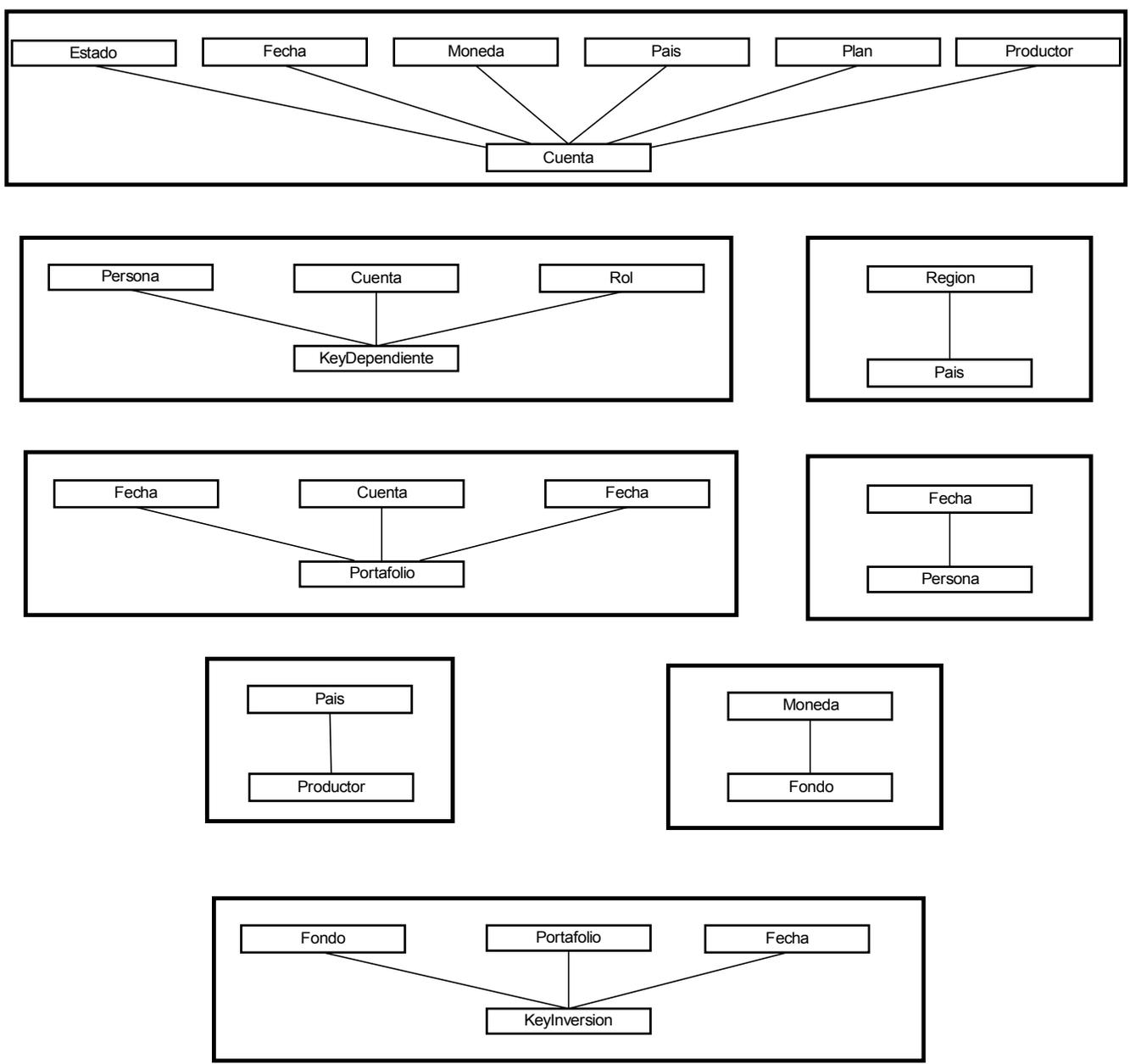


Figura 31. Regla IV: Primer orden entre niveles

Regla V: Identificación de orden entre niveles a partir de cardinalidad 1:N

A través de esta regla, se definen órdenes entre los niveles en base al conjunto de relaciones identificado en la *Regla III*, cuya cardinalidad sea 1:N. A continuación, se resumen los órdenes definidos:

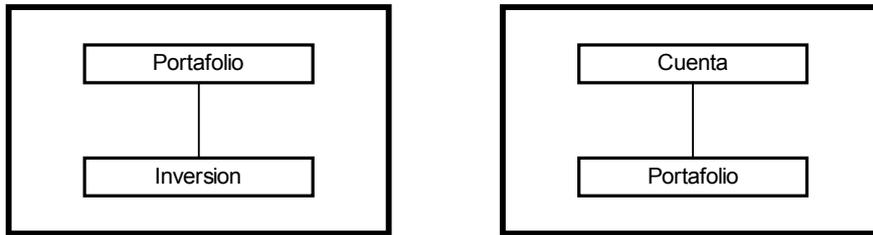


Figura 32. Regla V: Nuevo orden entre niveles

Regla VI: Identificación de orden entre niveles a partir de cardinalidad N:N

Dado que el modelo de dominio no cuenta con cardinalidades N:N, esta regla no se aplica.

Regla VII: Identificación de orden entre niveles a partir de relaciones 1:1

Esta regla plantea una relación de orden entre niveles representados por clases que mantengan una relación 1:1 con totalidad. En el modelo de dominio en estudio, todos los niveles cuyas clases mantienen una relación 1:1 con totalidad en una de ellas, mantienen también una relación 1:1 con totalidad en la otra clase, por lo tanto no existe un orden entre estos niveles sino que son niveles en un mismo nivel. Estos casos se presentan en el cuadro siguiente.

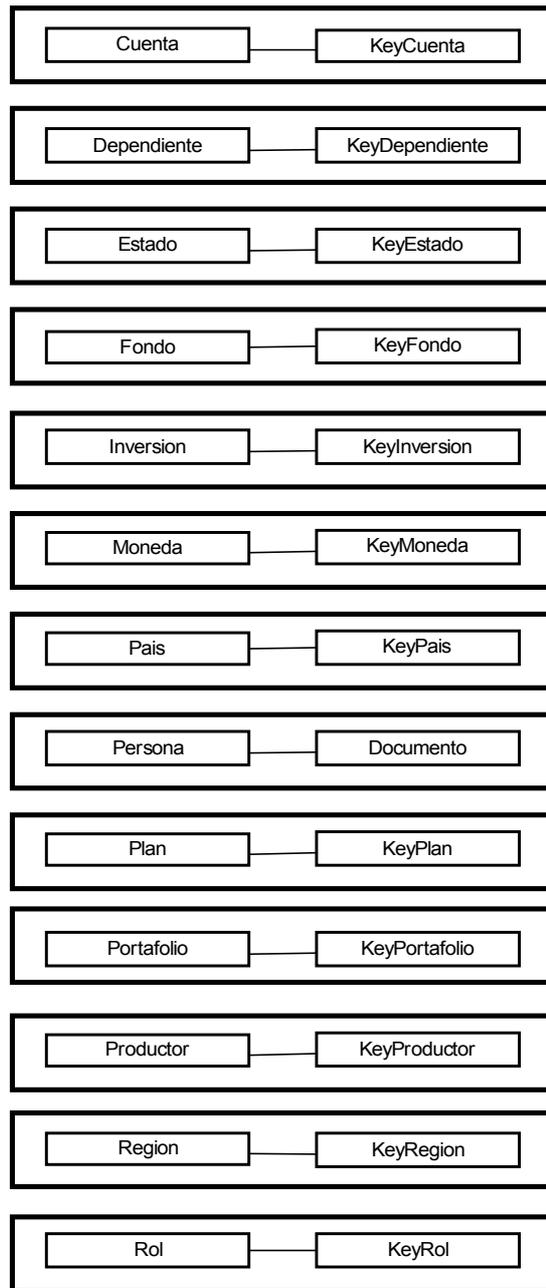


Figura 33. Regla VII: Tercer regla de orden entre niveles

Regla VIII: Identificación de orden entre niveles a partir de cardinalidad 1:1

Dado que el modelo de dominio no cuenta con relaciones de cardinalidad 1:1 (sin totalidad), esta regla no se aplica en este caso de estudio.

Regla IX: Categorización de subclases (a partir de subclases disjuntas)

La aplicación de esta regla implica definir un nuevo nivel correspondiente a la categorización definida por superclases; este será un nivel superior de los niveles representados por clases que son subclases de las anteriores.

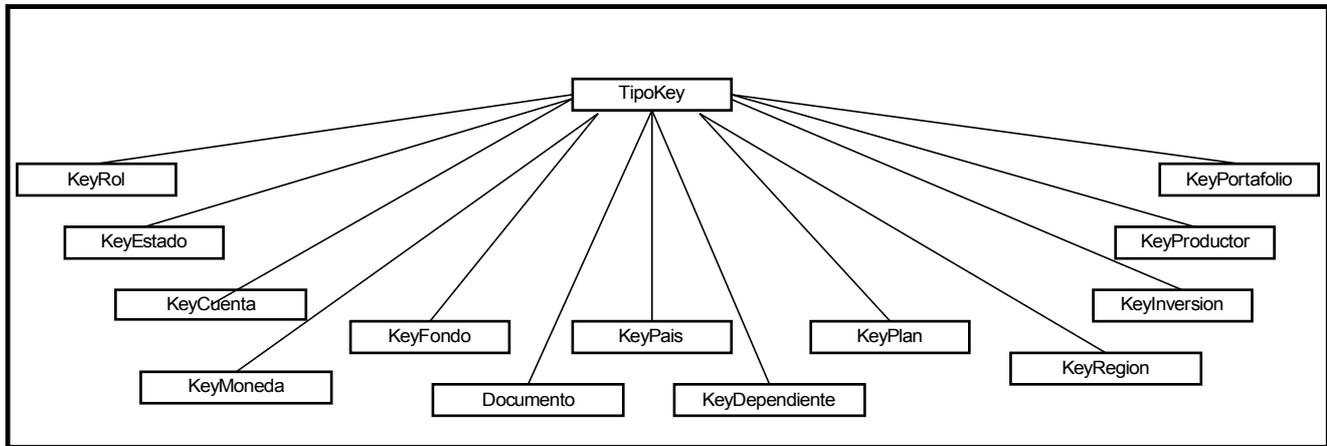


Figura 34. Regla IX: Categorización de subclases

Regla X: Identificación de dimensiones en CMDM

La aplicación de esta regla define cuáles serán las dimensiones del modelo multidimensional. Para el caso de estudio en cuestión, se obtuvo una única dimensión. A continuación, se presenta el resultado obtenido hasta el momento, en el cual se observa la dimensión KeyInversion y el orden en sus niveles superiores.

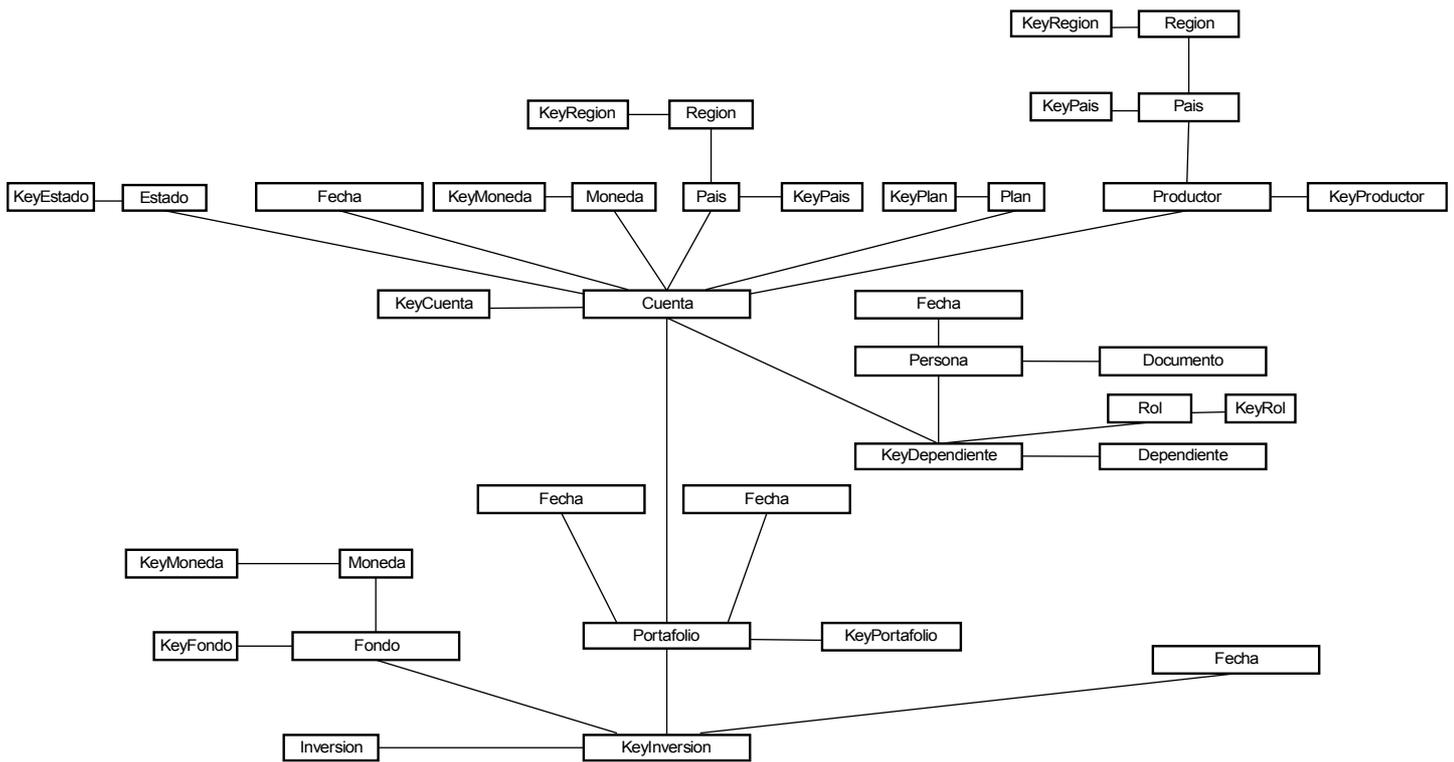


Figura 35. Regla X: Dimensiones

Para mayor claridad en la Figura 35 y dado que no es relevante en el modelo multidimensional, no se agregó el nivel TipoKey definido en la Regla IX.

Regla XI: Identificación de relaciones dimensionales en CMDM

A través de esta regla, se definen tres dimensiones en el modelo multidimensional: Fondo, Portafolio y Fecha como se muestra en la Figura 36.

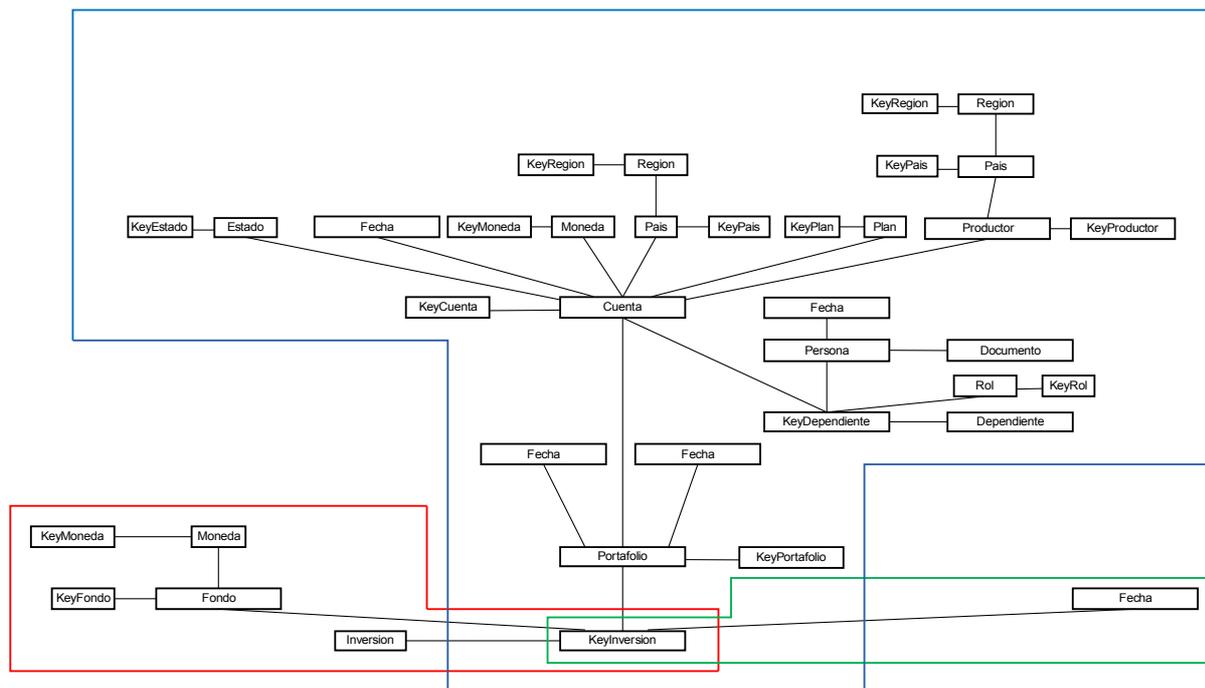


Figura 36. Regla XI: Relación Dimensional

Resultado obtenido – Fase 3

Se obtuvo la relación dimensional *Activos* con tres dimensiones:

1. Tiempo
2. Fondo
3. Portafolio

Las primeras tres dimensiones se han identificado luego de la aplicación de las reglas de MD4DW y teniendo en cuenta los requerimientos planteados.

La dimensión Tiempo está compuesta por los niveles Inversión y Fecha. Tal como se especificó en los requerimientos, la fecha sobre la cual se medirá la dimensión tiempo corresponde a la fecha de evaluación de las inversiones.

Dimensión Tiempo

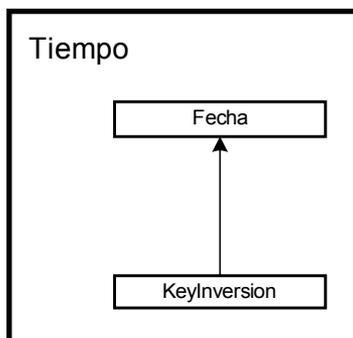


Figura 37. Dimensión Tiempo

Dimensión Fondo

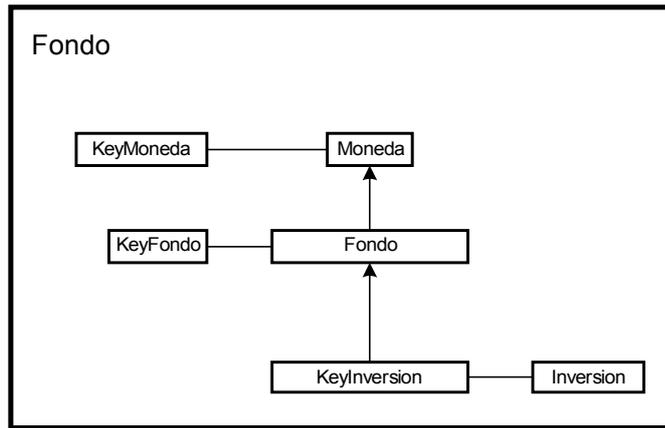


Figura 38. Dimensión Fondo

Dimensión Portafolio

Portafolio

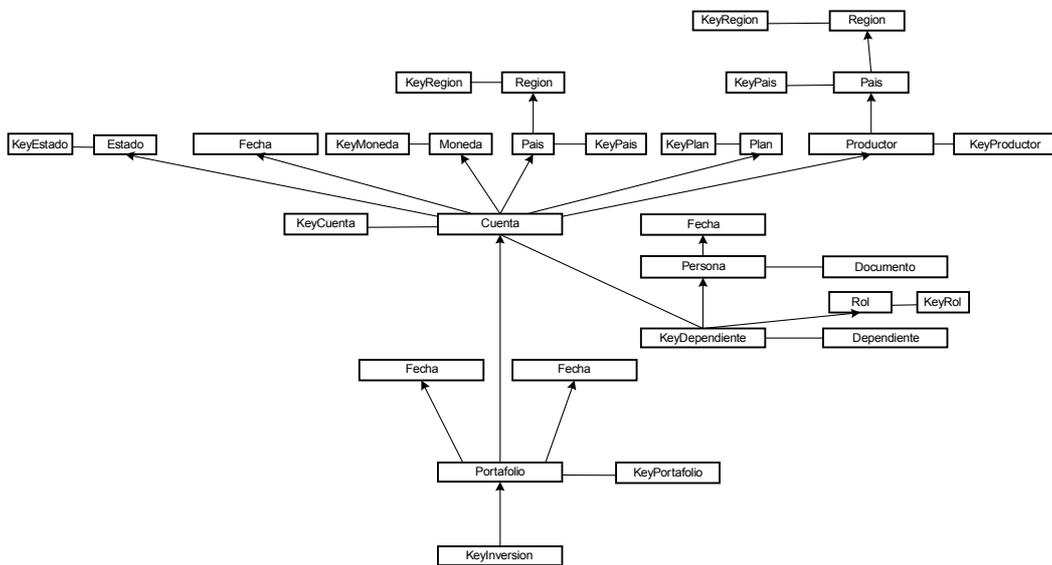


Figura 39. Dimensión Portafolio

Relación dimensional Activos

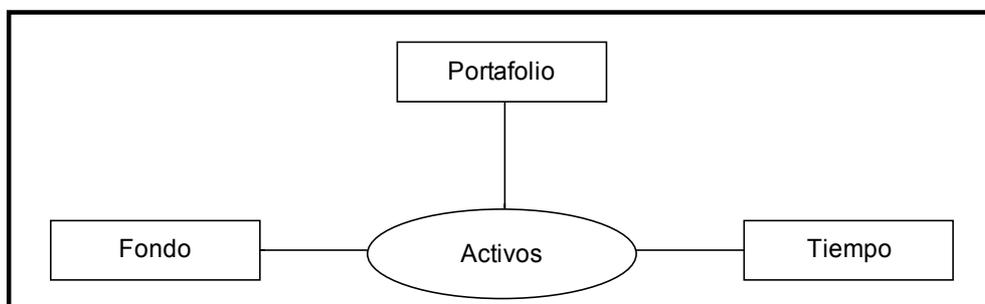


Figura 40. Relación dimensional Activos

Segunda ejecución – Ontología sin concepto *Clave*

Fase 1 - Entender y documentar la realidad

Esta fase no difiere de la Fase 1 de la ejecución anterior, ya que la realidad no ha cambiado.

Fase 2 - Modelado del dominio

Se representa la realidad antes mencionada mediante una ontología en la cual no se modelan (como entidades) las claves de los conceptos. Esta decisión se ha tomado debido a la dificultad encontrada anteriormente en la representación y aplicación de la metodología cuando se representan las claves.

La Figura 41 presenta la ontología desarrollada, mucho más simple y fácil de comprender que el caso anterior (aunque no es tan completa):

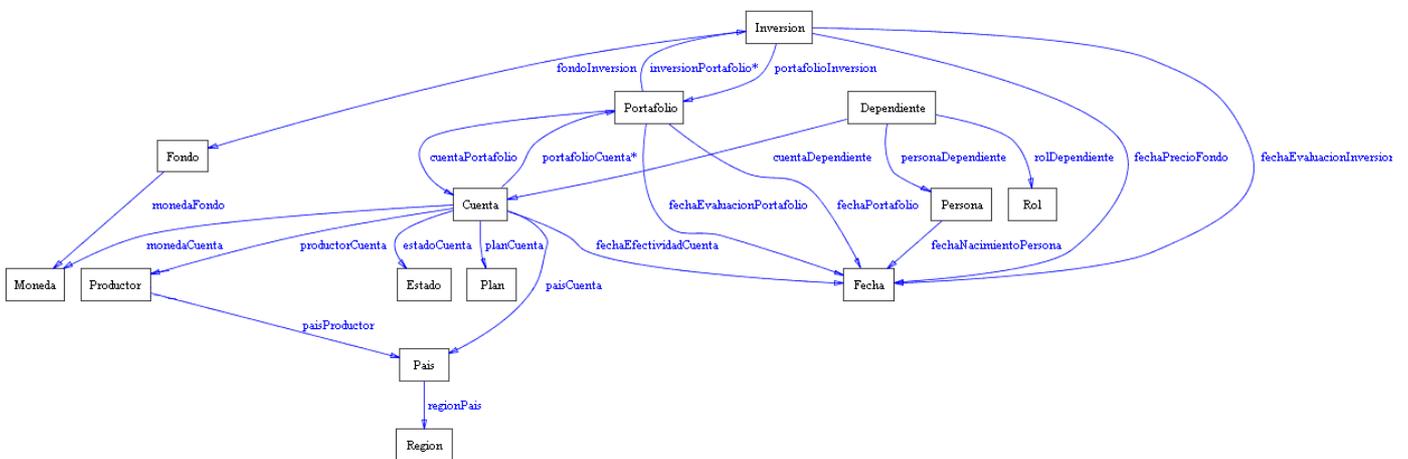


Figura 41. Ontología sin conceptos de claves

Además de la ontología se genera una nueva matriz de requerimientos ya que difiere de la matriz anterior debido a que no aparece el concepto “Documento” ni la relación “personaDocumento” porque este concepto representaba la clave de una Persona.

Concepto	RQ1	RQ2
Clases		
Cuenta		√
Dependiente		√
Estado		√
Fecha	√	√
Fondo	√	√
Inversión	√	√
Moneda	√	√
País		√
Persona		√
Plan		√
Portafolio	√	√
Productor		√

Región		√
Rol		√
Relaciones		
cuentaPortafolio		√
dependienteCuenta		√
estadoCuenta		√
fechaEfectividadCuenta		√
fechaEvaluacionInversion	√	√
fechaEvaluacionPortafolio	√	√
fechaNacimientoPersona		√
fechaPortafolio	√	√
fechaPrecioFondo	√	√
fondoInversion	√	√
inversionPortafolio	√	√
monedaCuenta		√
monedaFondo	√	√
paisCuenta		√
paisProductor		√
personaDependiente		√
planCuenta		√
portafolioCuenta		√
portafolioInversion	√	√
productorCuenta		√
regionPais		√
rolDependiente		√

Fase 3 - Construcción del modelo conceptual

Utilizando el mismo documento de Metadata de Conceptos (sin el concepto *Documento*), la nueva matriz de requerimientos y la nueva ontología del dominio, se procedió a aplicar las reglas definidas por la metodología MD4DW.

Regla I: Clases a mapearse como niveles en CMDM

Se identificaron las clases identificadas en la matriz de trazabilidad para los dos requerimientos del caso de estudio

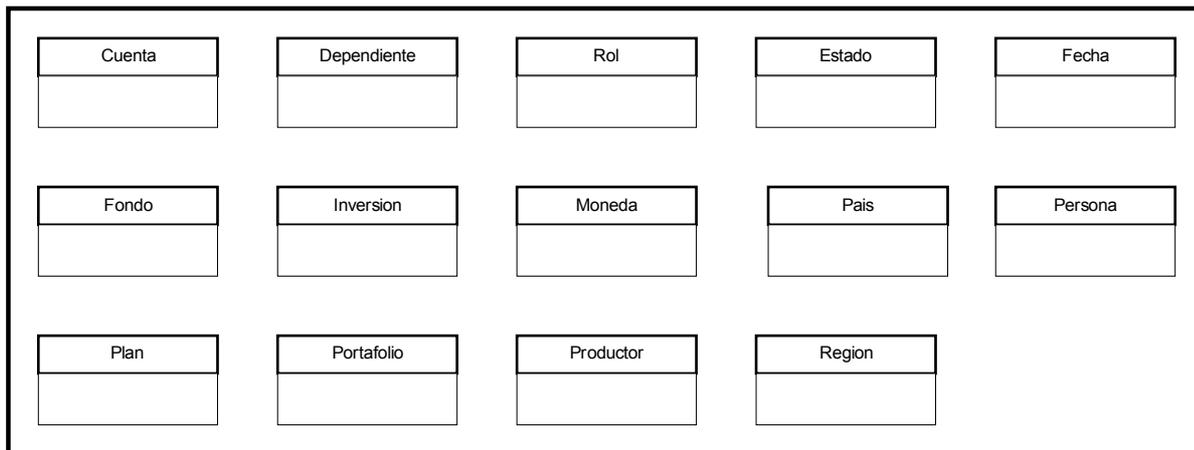


Figura 42. Regla I: Clases candidatas a niveles

Regla II: Definición de niveles en CMDM

El resultado de la aplicación de esta regla define el conjunto de niveles indicando para cada uno su nombre y sus atributos simples.

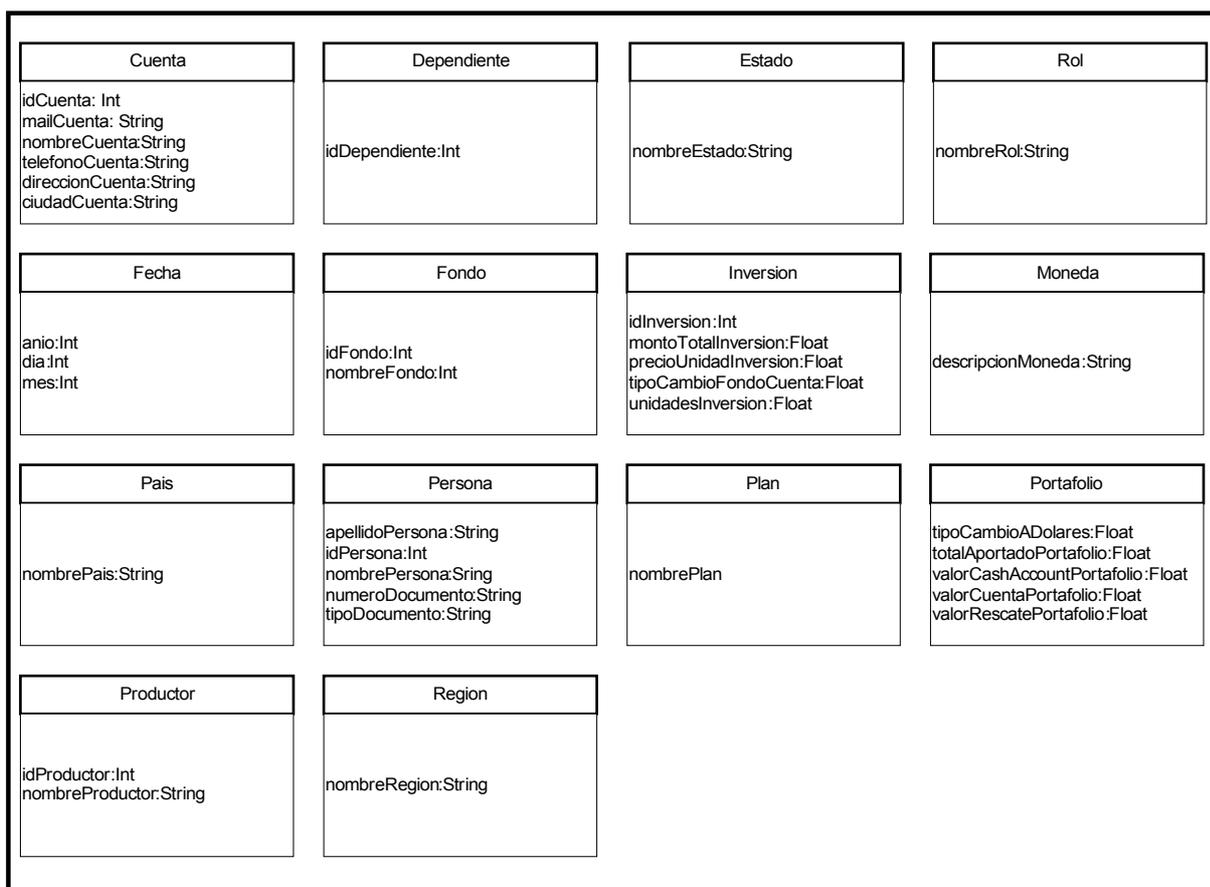


Figura 43. Regla II: Conjunto de niveles con atributos

Regla III: Definición de cardinalidades

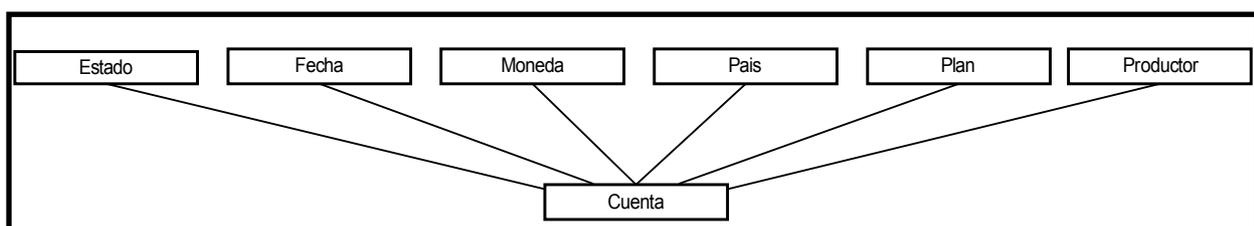
Como lo plantea la regla, se identificaron todas las relaciones entre clases candidatas a niveles, y se definió para cada relación su cardinalidad.

Resultado de la aplicación de esta tercera regla:

Relación	Dominio	Rango
estadoCuenta	N	1
fechaEfectividadCuenta	N	1
monedaCuenta	N	1
paisCuenta	N	1
planCuenta	N	1
portafolioCuenta	1	N
productorCuenta	N	1
monedaFondo	N	1
fechaPrecioFondo	N	1
regionPais	N	1
fechaNacimientoPersona	N	1
cuentaPortafolio	N	1
fechaEvaluacionPortafolio	N	1
fechaPortafolio	N	1
inversionPortafolio	N	1
paisProductor	N	1
cuentaDependiente	N	1
personaDependiente	N	1
rolDependiente	N	1
fechaEvaluacionInversion	N	1
fondoInversion	N	1
portafoliInversion	N	1

Regla IV: Identificación de orden entre niveles a partir de cardinalidad N:1

Una vez identificadas las relaciones con cardinalidad N:1 de la *Regla III*, se establece un primer orden entre los niveles definidos en la *Regla II*:



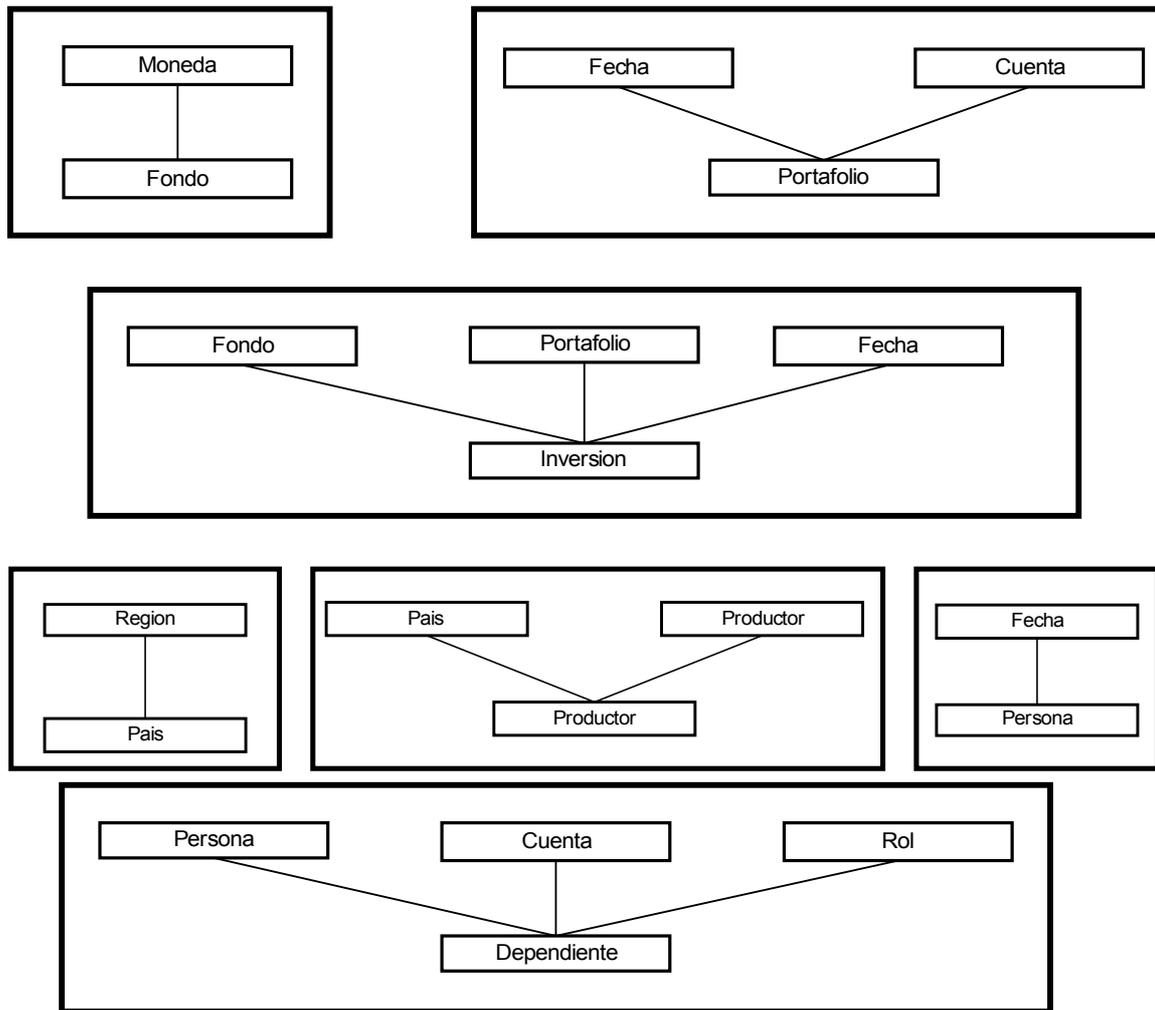


Figura 44. Regla IV: Primer orden entre niveles

Regla V: Identificación de orden entre niveles a partir de cardinalidad 1:N

A través de esta regla, se definen órdenes entre los niveles en base al conjunto de relaciones identificado en la *Regla III*, cuya cardinalidad sea 1:N. A continuación, se resumen los órdenes definidos:

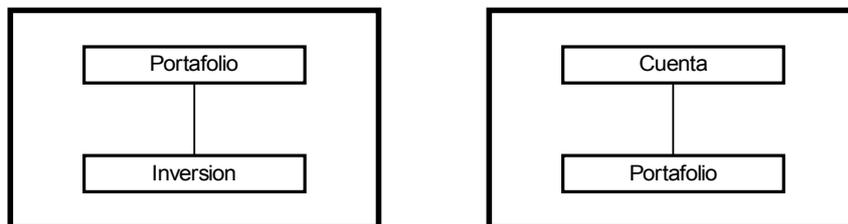


Figura 45. Regla V: Nuevo orden entre niveles

Regla VI: Identificación de orden entre niveles a partir de cardinalidad N:N

Regla VII: Identificación de orden entre niveles a partir de relaciones 1:1

Regla VIII: Identificación de orden entre niveles a partir de cardinalidad 1:1

Dado que el modelo de dominio no cuenta con cardinalidades N:N, ni 1:1 estas reglas no se aplican.

Regla IX: Categorización de subclases (a partir de subclases disjuntas)

Dado que en la ontología no existen especializaciones, por lo que esta regla no es aplicada.

Regla X: Identificación de dimensiones en CMDM

La aplicación de esta regla define cuáles serán las dimensiones del modelo multidimensional. Para el caso de estudio en cuestión, se obtuvo una únicamente dimensión. A continuación, se presenta el resultado obtenido hasta el momento, en el cual se observa la dimensión Inversión y el orden en sus niveles superiores.

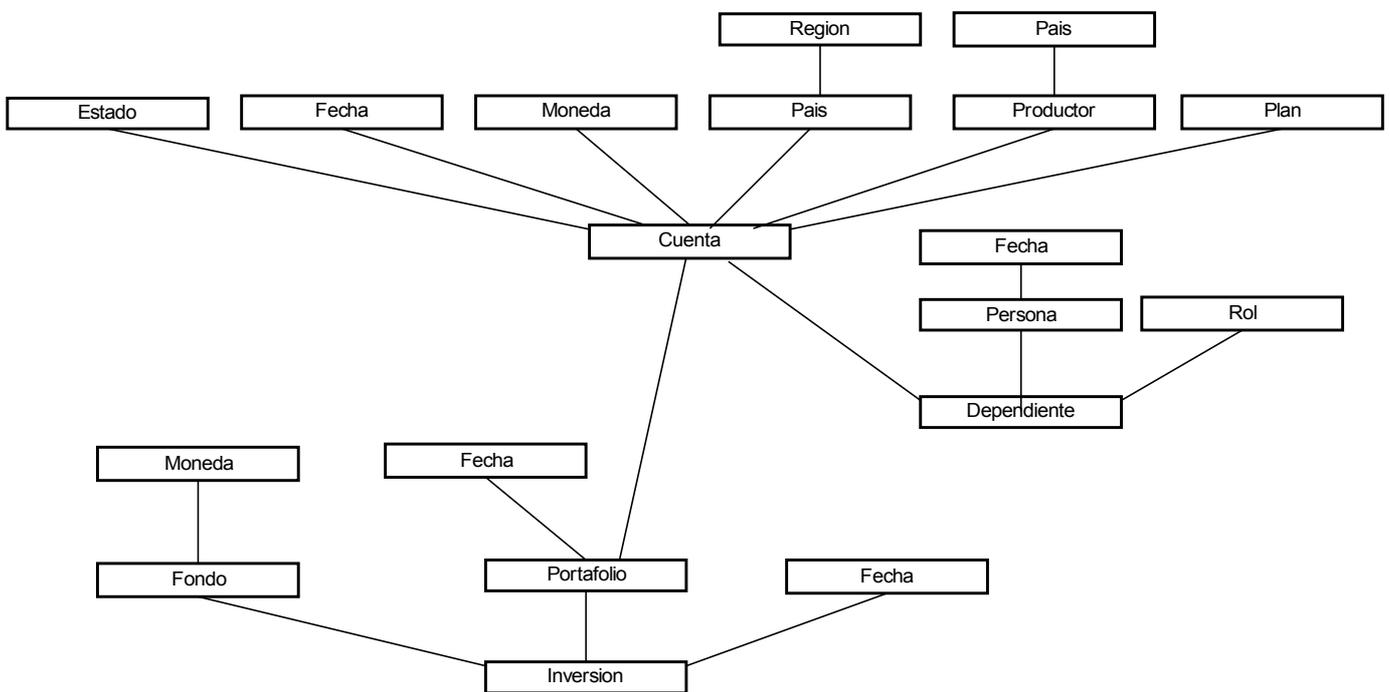


Figura 46. Regla X: Dimensiones

Regla XI: Identificación de relaciones dimensionales en CMDM

A través de esta regla, se definen tres relaciones multidimensionales en el modelo multidimensional: Fondo, Portafolio y Fecha como se presentada en la Figura 47.

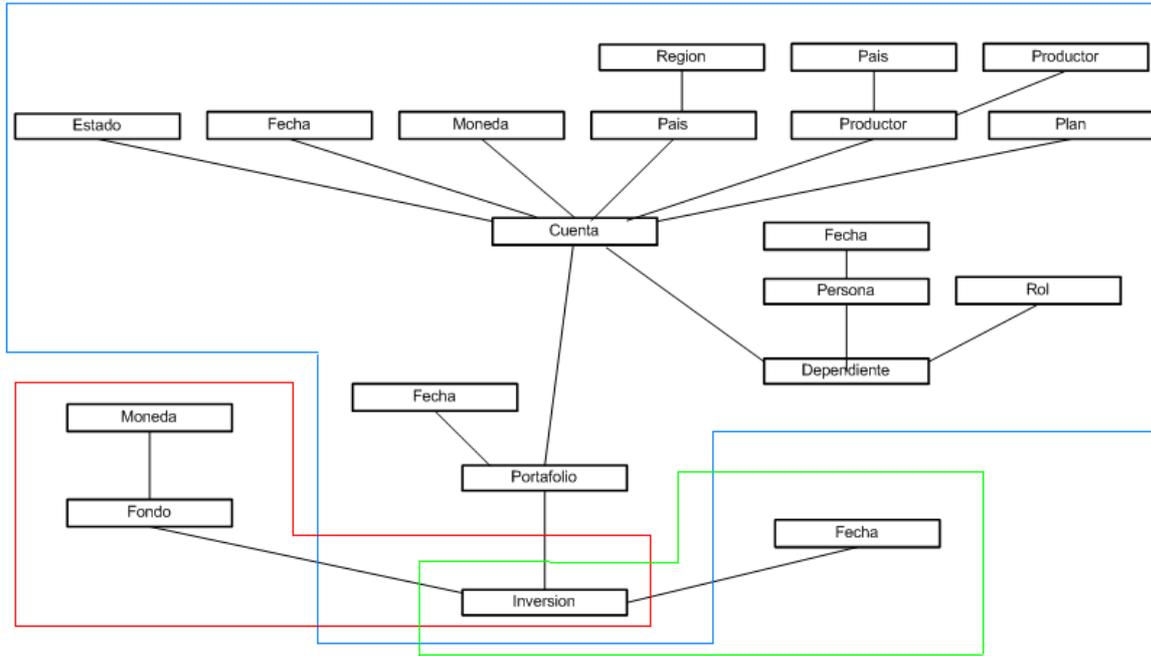


Figura 47. Regla XI: Relación Dimensional

Resultado obtenido – Fase 3

Al igual que en la primera ejecución del caso de estudio, se obtuvo la relación dimensional *Activos* con tres dimensiones:

- 1- Tiempo
- 2- Fondo
- 3- Portafolio

Las tres dimensiones se han identificado luego de la aplicación de las reglas de MD4DW y teniendo en cuenta los requerimientos planteados.

Dimensión Tiempo

La dimensión Tiempo está compuesta por los niveles *Inversión* y *Fecha* ().

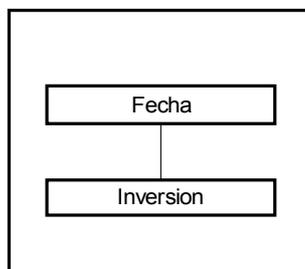


Figura 48. Dimensión Tiempo

Dimensión Fondo

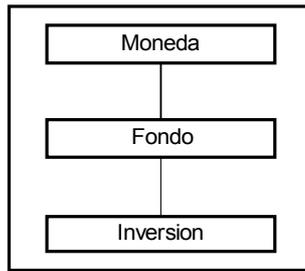


Figura 49. Dimensión Fondo

Dimensión Portafolio

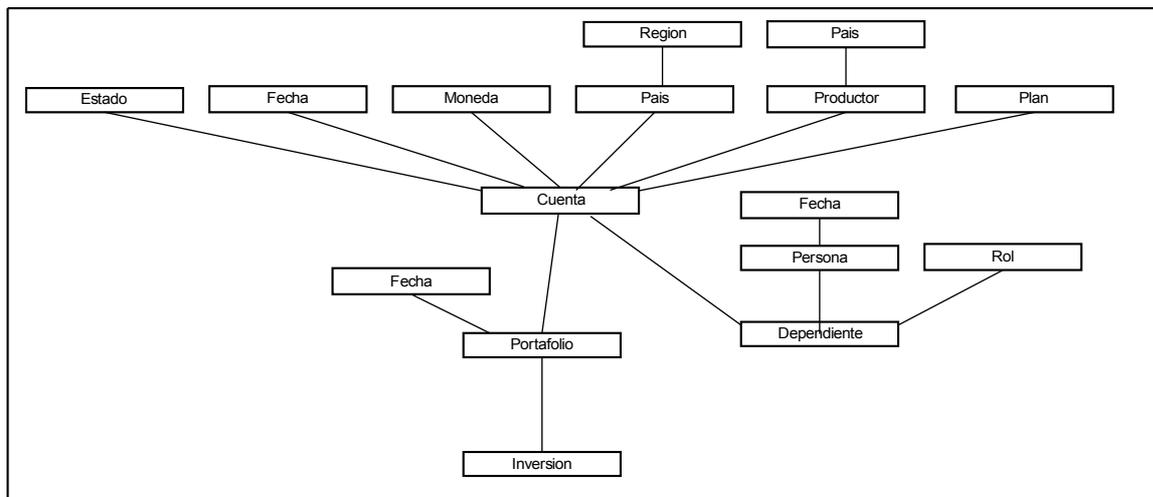


Figura 50. Dimensión Portafolio

Relación dimensional Activos

Como vemos en la Figura 51 , el resultado obtenido es el mismo que el resultado final utilizando la ontología compleja.

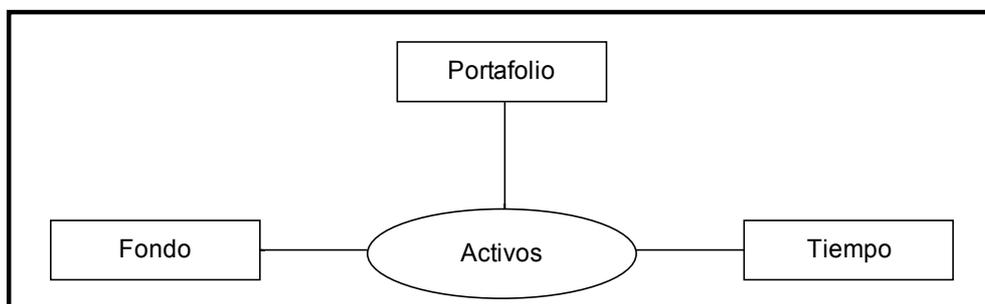


Figura 51. Relación dimensional Activos

Aplicación de la guía unificada (Caso de estudio cuentas fondos)

En las siguientes secciones se describe el desarrollo de un Caso de Estudio en el cual se aplica una guía definida para obtener el diseño y la carga de un DW partiendo de una realidad y los requerimientos sobre la misma.

En esta ejecución, la cual es una continuación de casos de estudios anteriores (ver Análisis del contexto de trabajo - Caso de Estudio), se puede apreciar la aplicación completa de las cinco fases de la guía propuesta.

Se describen a continuación las cinco fases definidas:

Fase 1 – Aplicación de la Fase 1 de MD4DW.

Salida: - Metadata de conceptos
- Descripción de los requerimientos

Fase 2 – Aplicación de la Fase 2 de MD4DW

Entrada: - Metadata de conceptos
- Descripción de los requerimientos

Salida: - Ontología de dominio
- Matriz de requerimientos

Fase 3 – Aplicación de la Fase 3 de MD4DW

Entrada: - Metadata de conceptos
- Matriz de requerimientos
- Ontología de dominio

Salida: - Modelo conceptual multidimensional en CMDM

Fase 4 – Aplicación del algoritmo Onto2rdb

Entrada: - Ontología de dominio

Salida: - ODS

Fase 5 – Aplicación del algoritmo de Larrañaga

Entrada: - Modelo conceptual multidimensional en CMDM
- Descripción de los requerimientos
- ODS

Salida: - Diseño de DW relacional
- Diseño de la carga del DW relacional

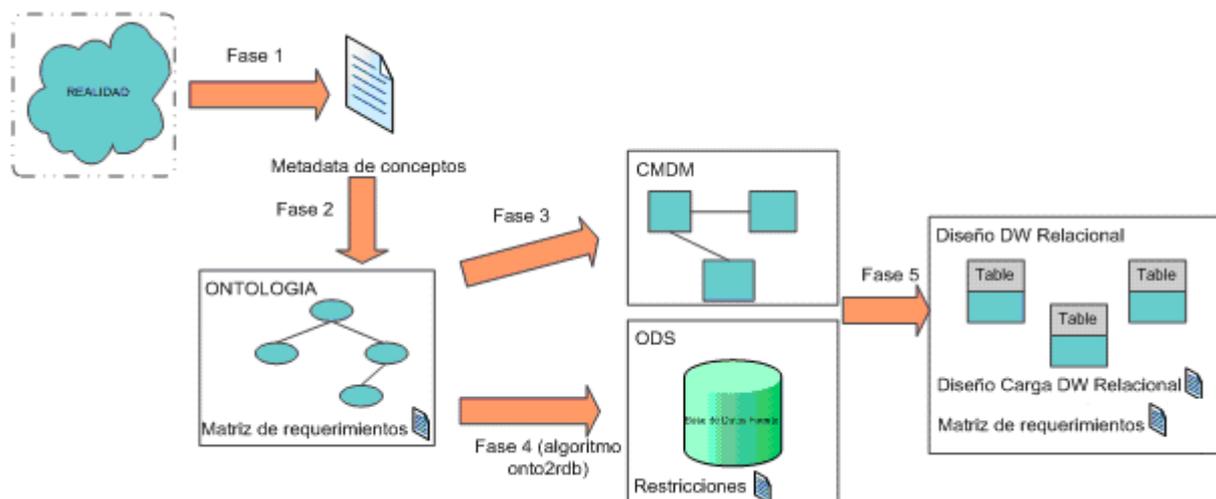


Figura 52. Fases de la guía completa para la construcción de un DW

Aplicación de la guía para la construcción de un DW

Fase 1 - Entender y documentar la realidad

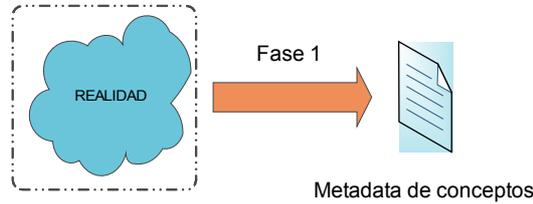


Figura 53. Fase 1

No hay cambios en los requerimientos, por lo que la Metadata de conceptos no es necesario reescribirla.

Fase 2 - Modelado del dominio – Ontología sin modelar claves ni fechas.

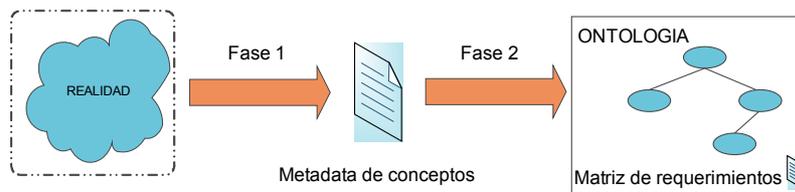


Figura 54. Fase 2

Se representa la realidad antes mencionada mediante una ontología en la cual no se modelan (como entidades) las claves de los conceptos. Esta decisión se ha tomado debido a la dificultad encontrada anteriormente en la representación y aplicación de la metodología cuando se representan las claves.

Además, se genera la siguiente matriz de requerimientos:

Concepto	RQ1	RQ2
Clases		
Cuenta		√
Dependiente		√
Estado		√
Fondo	√	√
Inversión	√	√
Moneda	√	√
País		√
Persona		√
Plan		√
Portafolio	√	√
Productor		√
Región		√
Rol		√
Relaciones		

cuentaPortafolio		√
dependienteCuenta		√
estadoCuenta		√
fondoInversion	√	√
inversionPortafolio	√	√
monedaCuenta		√
monedaFondo	√	√
paisCuenta		√
paisProductor		√
personaDependiente		√
planCuenta		√
portafolioCuenta		√
portafolioInversion	√	√
productorCuenta		√
regionPais		√
rolDependiente		√

Las fechas no fueron consideradas como conceptos, sino como atributos, definiendo luego cuáles pertenecerían a la dimensión Tiempo del Datawarehouse. Esta decisión fue tomada en base a los resultados de los casos de estudio anteriores. Si cada fecha es considerada como un concepto, se pierde luego el enfoque sobre cuáles de ellas debe pertenecer a la dimensión Tiempo.

Dado que MD4DW recomienda asumir que el concepto Tiempo siempre existe en todo DW, no se trata este concepto en esta etapa y se incluye nuevamente al obtener las dimensiones al final de la metodología.

La Figura 55 presenta la ontología desarrollada, aún más simple y considerando sólo aquellos conceptos relevantes (según los requerimientos) para el diseño del DW:

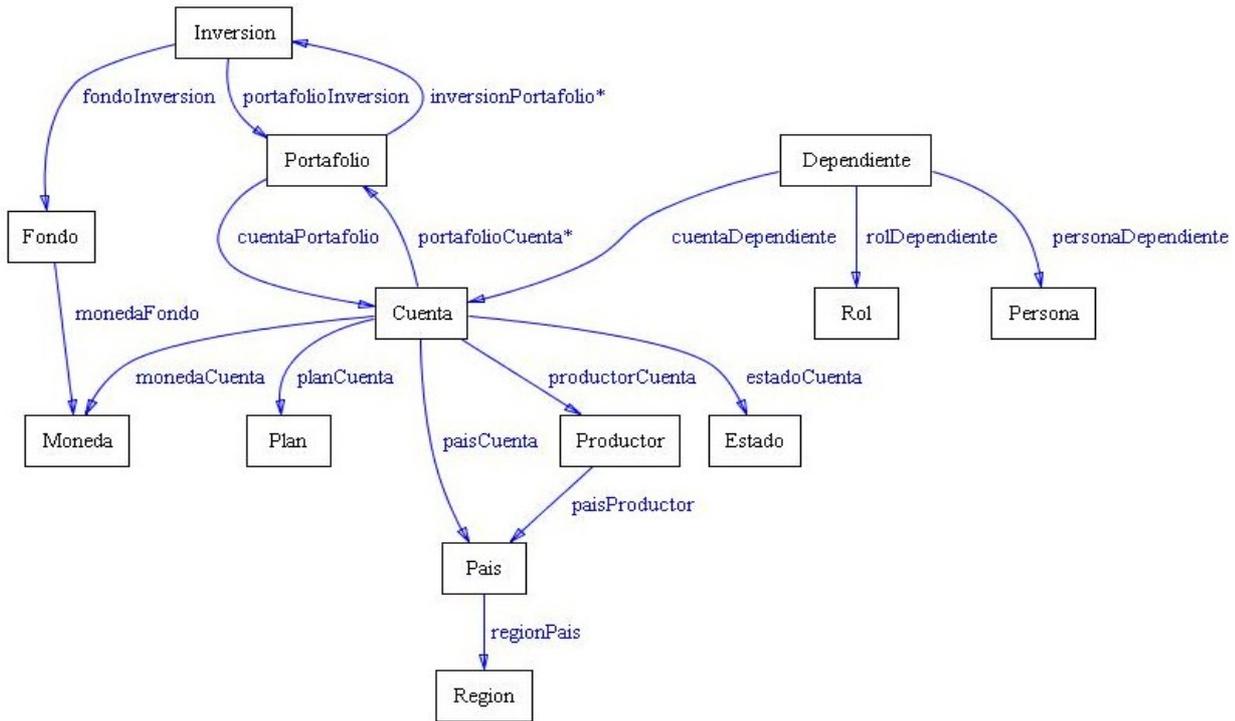


Figura 55. Ontología obtenida en el Caso de Estudio

Fase 3 - Construcción del modelo conceptual

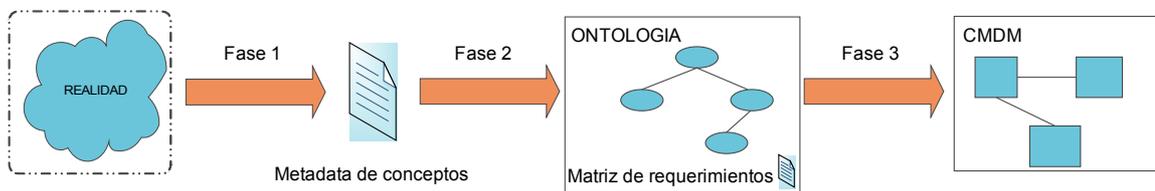


Figura 56. Fase 3

Una vez obtenido el documento de Metadata de Conceptos, la matriz de requerimientos y la ontología del dominio, se procedió a aplicar las reglas definidas por la metodología MD4DW.

Regla I: Clases a mapearse como niveles en CMDM

Cuenta	Dependiente	Rol	Estado	Persona
Fondo	Inversion	Moneda	Pais	Region
Plan	Portafolio	Productor		

Figura 57. Regla I: Clases a mapearse como niveles

Regla II: Definición de niveles en CMDM

El resultado de la aplicación de esta regla define el conjunto de niveles indicando para cada uno su nombre y sus atributos simples.

<p>Cuenta</p> <p>numeroCuenta: String mailCuenta: String nombreCuenta: String telefonoCuenta: String direccionCuenta: String ciudadCuenta: String fechaEfectividad: Date</p>	<p>Dependiente</p>	<p>Estado</p> <p>nombreEstado:String</p>	<p>Rol</p> <p>nombreRol:String</p>
<p>Portafolio</p> <p>tipoCambioADolares: Float totalAportado: Float valorCashAccount: Float valorCuenta: Float valorRescate: Float fechaEvaluacionPortafolio: Date fechaPortafolio: Date</p>	<p>Fondo</p> <p>numeroFondo: String nombreFondo: String</p>	<p>Inversion</p> <p>monto: Float precio: Float tipoCambioFondoCuenta: Float unidades: Float fechaPrecioFondo: Date fechaEvaluacion: Date</p>	<p>Moneda</p> <p>descripcionMoneda:String</p>
<p>Pais</p> <p>nombrePais:String</p>	<p>Persona</p> <p>apellidoPersona:String nombrePersona:String numeroDocumento:String tipoDocumento:String fechaNacimiento:Date</p>	<p>Plan</p> <p>nombrePlan :String</p>	<p>Region</p> <p>nombreRegion:String</p>
<p>Productor</p> <p>numeroContratoProductor: String nombreProductor:String</p>			

Figura 58. Clases a mapearse como niveles

Regla III: Definición de cardinalidades

Como lo plantea la regla, se identificaron todas las relaciones entre clases candidatas a niveles, y se definió para cada relación su cardinalidad. Resultado de la aplicación de esta tercera regla:

Relación	Dominio	Rango
estadoCuenta	N	1
monedaCuenta	N	1
paisCuenta	N	1
planCuenta	N	1
portafolioCuenta	1	N
productorCuenta	N	1
monedaFondo	N	1
regionPais	N	1
cuentaPortafolio	N	1
inversionPortafolio	1	N
paisProductor	N	1
cuentaDependiente	N	1
personaDependiente	N	1
rolDependiente	N	1
fondoInversion	N	1
portafolioInversion	N	1

Regla IV: Identificación de orden entre niveles a partir de cardinalidad N:1

Una vez identificadas las relaciones con cardinalidad N:1 de la *Regla III*, se establece un primer orden entre los niveles definidos en la *Regla II*:

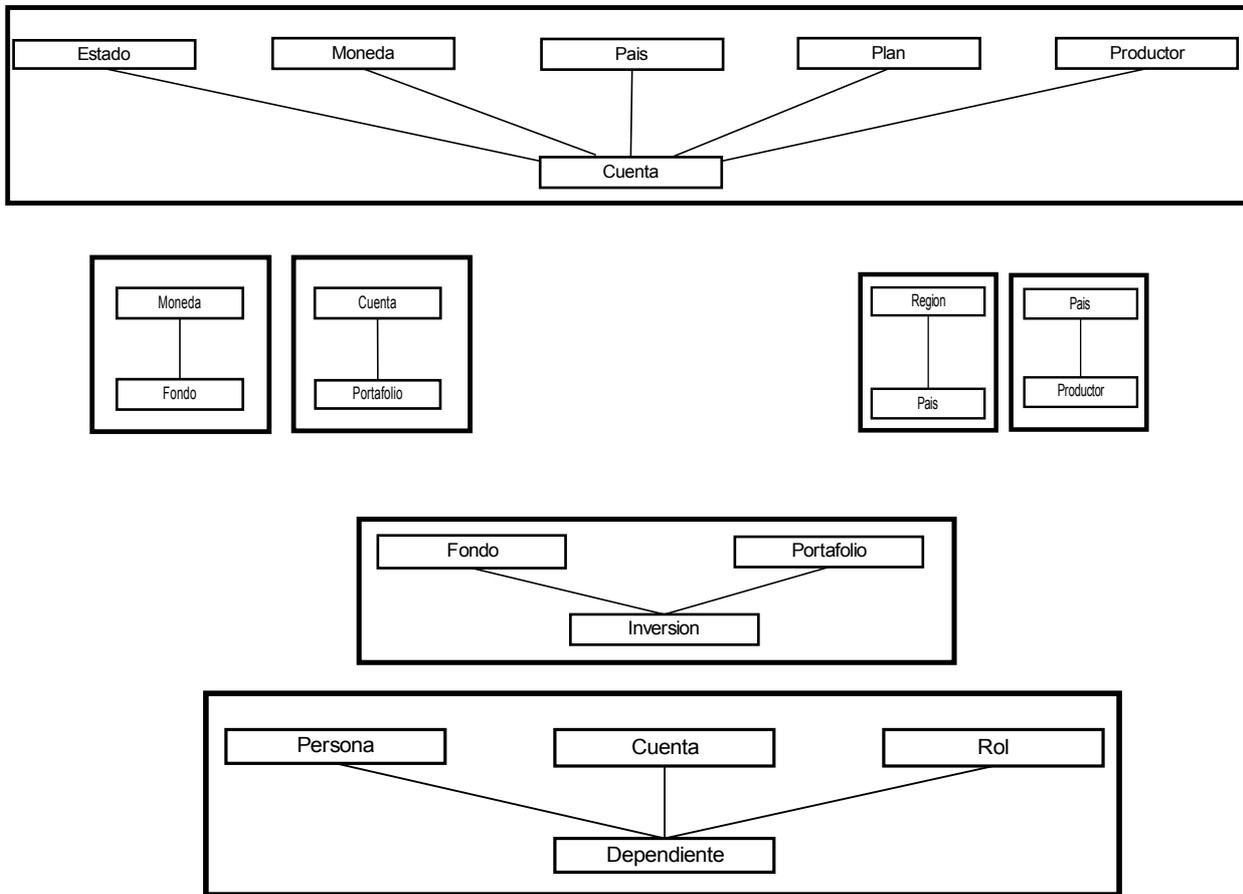


Figura 59. Regla IV: Primer orden entre niveles

Regla V: Identificación de orden entre niveles a partir de cardinalidad 1:N

A través de esta regla, se definen órdenes entre los niveles en base al conjunto de relaciones identificado en la *Regla III*, cuya cardinalidad sea 1:N. A continuación, se resumen los órdenes definidos:

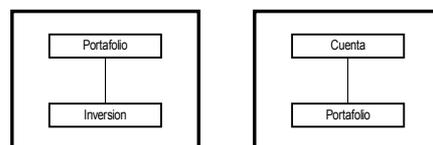


Figura 60. Regla V: Nuevo orden entre niveles

Regla VI: Identificación de orden entre niveles a partir de cardinalidad N:N

Regla VII: Identificación de orden entre niveles a partir de relaciones 1:1

Regla VIII: Identificación de orden entre niveles a partir de cardinalidad 1:1

Dado que el modelo de dominio no cuenta con relaciones de cardinalidad N:N ni 1:1, estas reglas no se aplican en este caso de estudio.

Regla IX: Categorización de subclases (a partir de subclases disjuntas)

Dado que en la ontología no existen especializaciones, esta regla no es aplicada.

Regla X: Identificación de dimensiones en CMDM

La aplicación de esta regla define cuáles serán las dimensiones del modelo multidimensional. Para el caso de estudio en cuestión, se obtuvieron dos dimensiones, esto tomando la decisión de tener varias dimensiones y no de mantener una única dimensión con varias jerarquías. Esta decisión de diseño fue tomada en base a los requerimientos planteados. A continuación, se presenta el resultado obtenido hasta el momento, en el cual se observan las dimensiones *Fondo* y *Portafolio*, y el orden en sus niveles superiores.

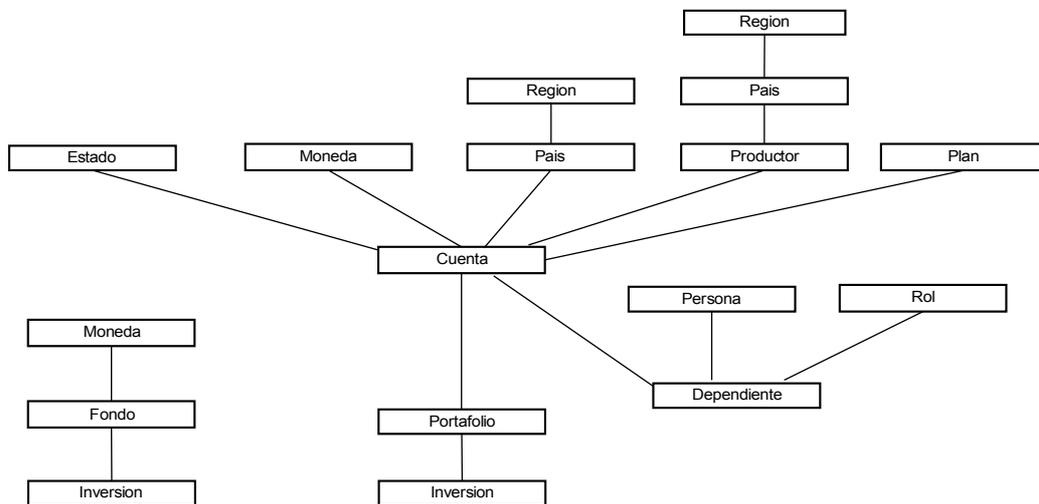


Figura 61. Regla X: Dimensiones en CMDM

Regla XI: Identificación de relaciones dimensionales en CMDM

A través de esta regla, se define la relación dimensional *Activos* en el modelo multidimensional:

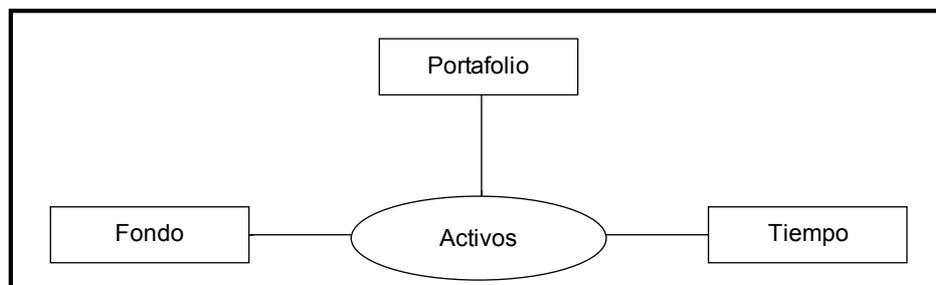


Figura 62. Regla XI: Relación Dimensional del Caso de Estudio Resultado obtenido – Fase 3

Se obtuvo la relación dimensional *Activos* con tres dimensiones:

- 1- Fondo
- 2- Portafolio
- 3- Tiempo

Las primeras dos dimensiones se han identificado luego de la aplicación de las reglas de MD4DW y teniendo en cuenta los requerimientos planteados.

Dimensión Fondo

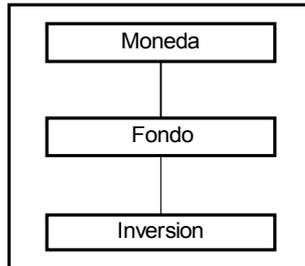


Figura 63. Dimensión Fondo

Dimensión Portafolio

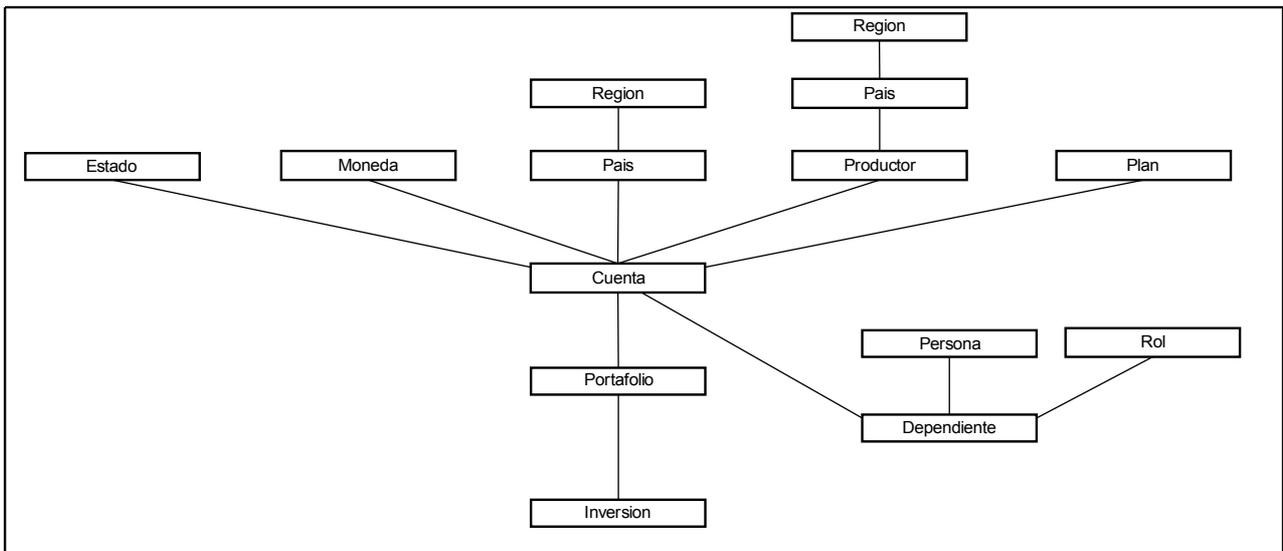


Figura 64. Dimensión Portafolio

Dimensión Tiempo: Esta dimensión, además de representar uno de los requerimientos, se agregó posteriormente por el supuesto de MD4DW de que siempre estará presente en cualquier DW. Dicha dimensión está compuesta por los niveles *Inversión* y *Fecha*. Tal como se especificó en los requerimientos, la fecha sobre la cual se medirá la dimensión tiempo corresponde a la fecha de evaluación de las inversiones.

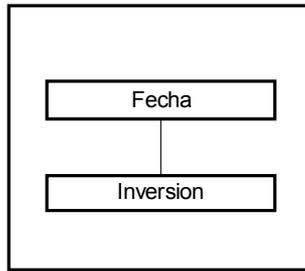


Figura 65. Dimensión Tiempo

Relación dimensional Activos

Las tres dimensiones antes mencionadas se relacionan mediante la relación dimensional *Activos*.

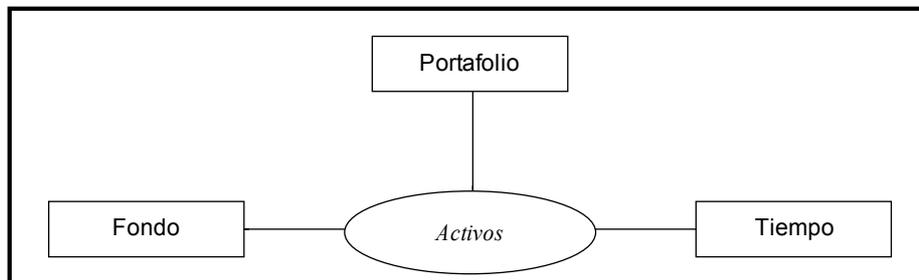


Figura 66. Relación dimensional *Activos*

Fase 4 - Construcción del ODS (algoritmo ONTO2RDB)

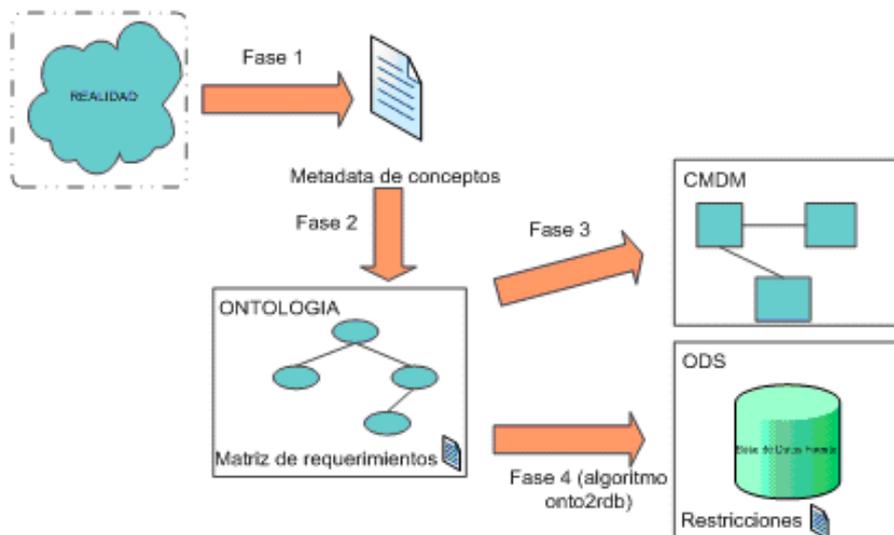


Figura 67. Fase 4

Se aplica el algoritmo *Onto2rdb* definido en este trabajo para generar el ODS a partir del cual se realizarán los mapeos con el esquema del modelo conceptual expresado en CMDM

1. Nuevo esquema

Se genera el esquema de base de datos vacío en Microsoft Access. La elección de la herramienta fue únicamente para seleccionar una aplicación de uso habitual, el resultado es independiente de la misma.

2. Una tabla por cada clase

Se genera una tabla por cada una de las clases que participan en la ontología, especificando para cada una sus atributos simples.

Se decidió definir claves sustitutas autonómicas en cada tabla para no trabajar con claves de tipo texto, ni con claves compuestas, casos presentes en el caso de estudio.

Para no perder las restricciones de unicidad que surgen de las claves naturales, se definieron restricciones que aseguran el cumplimiento de dicha unicidad.

A continuación, se detallan las tablas generadas y sus atributos. Los campos con cursiva indican las claves secundarias y los campos subrayados la clave primaria.

Cuenta (idCuenta, *numeroCuenta*, *nombreCuenta*, *direccionCuenta*, *ciudadCuenta*, *mailCuenta*, *telefonoCuenta*, *fechaEfectividad*)

Dependiente (idDependiente)

Rol (idRol, *nombreRol*)

Estado (idEstado, *nombreEstado*)

Fondo (idFondo, *nombreFondo*, *numeroFondo*)

Inversion (idInversion, *fechaPrecioFondo*, *unidades*, *precio*, *monto*, *tipoCambioFondoCuenta*, *fechaEvaluacion*)

Moneda (IdMoneda, *descripcionMoneda*)

Portafolio (idPortafolio, *fechaPortafolio*, *totalAportado*, *valorRescate*, *valorCuenta*, *valorCashAccount*, *tipoCambioADolares*, *fechaEvaluacionPortafolio*)

Plan (idPlan, *nombrePlan*)

Region (idRegion, *nombreRegion*)

Persona (idPersona, *apellidoPersona*, *nombrePersona*, *fechaNacimiento*, *numeroDocumento*, *tipoDocumento*)

Productor (idProductor, *nombreProductor*, *numeroContratoProductor*)

Pais (idPais, *nombrePais*)

3. Claves, relaciones 1 a 1 y relaciones n a 1

3.1. Claves

Todas las tablas tienen clave primaria definida debido a decisión de generar claves sustitutas autonómicas en el Paso 2, por lo que este paso de la iteración no se aplica en este ejemplo.

3.2. Relaciones n a 1

Para ejecutar este paso se aconseja la generación de una “Tabla de relaciones” (ordenadas alfabéticamente). Se puede reutilizar la tabla usada en la ejecución de la Fase 3 – Regla III.

Relación	Dominio	Rango	Cardinalidad (Dominio)	Cardinalidad (Rango)
cuentaDependiente	Dependiente	Cuenta	N	1
cuentaPortafolio	Portafolio	Cuenta	N	1
estadoCuenta	Cuenta	Estado	N	1
fondoInversion	Inversión	Fondo	N	1
inversionPortafolio	Portafolio	Inversión	1	N
monedaCuenta	Cuenta	Moneda	N	1
monedaFondo	Fondo	Moneda	N	1
paisCuenta	Cuenta	País	N	1
paisProductor	Productor	País	N	1
personaDependiente	Dependiente	Persona	N	1
planCuenta	Cuenta	Plan	N	1
portafolioCuenta	Cuenta	Portafolio	1	N
portafolioInversion	Inversión	Portafolio	N	1
productorCuenta	Cuenta	Productor	N	1
regionPaís	País	Región	N	1
rolDependiente	Dependiente	Rol	N	1

Para las 14 relaciones N:1 identificadas en la tabla de arriba se agregan las claves foráneas correspondientes. A continuación se presenta un ejemplo:

Dada la relación *cuentaDependiente*, cuyo dominio es *Dependiente* y el rango es *Cuenta*, se debe agregar en la tabla *Dependiente*, un campo llamado “*cuentaDependiente*” de tipo igual a al tipo de la clave de *Cuenta* (*idCuenta*).

Además se agrega la restricción referencial:

Dependiente.cuentaDependiente es clave foránea de *Cuenta.idCuenta*

De manera análoga se agrega un atributo para cada relación que cumple las condiciones descritas en el algoritmo (ver atributos en negrita).

Cuenta (*idCuenta*, *numeroCuenta*, *nombreCuenta*, *direccionCuenta*, *ciudadCuenta*, *mailCuenta*, *telefonoCuenta*, *fechaEfectividad*, ***estadoCuenta***, ***monedaCuenta***, ***paisCuenta***, ***planCuenta***, ***productorCuenta***)

Dependiente (*idDependiente*, ***cuentaDependiente***, ***rolDependiente***, ***personaDependiente***)

Rol (*idRol*, *nombreRol*)

Estado (*idEstado*, *nombreEstado*)

Fondo (*idFondo*, *nombreFondo*, *numeroFondo*, ***monedaFondo***)

Inversión (*idInversion*, *fechaPrecioFondo*, *unidades*, *precio*, *monto*, *tipoCambioFondoCuenta*, *fechaEvaluacion*, ***fondoInversion***, ***portafolioInversion***)

Moneda (*IdMoneda*, *descripcionMoneda*)

Portafolio (*idPortafolio*, *fechaPortafolio*, *totalAportado*, *valorRescate*, *valorCuenta*, *valorCashAccount*, *tipoCambioADolares*, *fechaEvaluacionPortafolio*,

cuentaPortafolio)

Plan (idPlan, nombrePlan)

Region (idRegion, nombreRegion)

Persona (idPersona, apellidoPersona, nombrePersona, fechaNacimiento, numeroDocumento, tipoDocumento)

Productor (idProductor, nombreProductor, numeroContratoProductor, **paisProductor**)

Pais (idPais, nombrePais, **regionPais**)

3.3. Relaciones 1 a 1

Dado que la ontología definida para el caso de estudio no cuenta con relaciones 1:1, este paso no es aplicado.

4. Relaciones n a n

Dado que la ontología definida para el caso de estudio no cuenta con relaciones N:N, este paso no es aplicado.

5. Jerarquías

Dado que la ontología definida para el caso de estudio no cuenta con jerarquías, este paso no es aplicado.

6. Log de restricciones no consideradas

Las restricciones que no fueron definidas y deberán ser implementadas en la transformación de los orígenes de datos al ODS son las siguientes:

Clase	Restricción	Descripción de la restricción
Inversión	Inversion.monto = Inversion.unidades * Inversion.precio	El monto de cada inversión es igual a las unidades de la inversión multiplicadas por el precio de las unidades de la inversión.
Portafolio	Portafolio.valorCuenta = « SUM(Inversion.monto * Inversion.tipoCambioFondoCuenta) WHERE Inversion.idPortafolio = Portafolio.idPortafolio» + Portafolio.valorCashAccount	El valor de cuenta de cada Portafolio es igual a la suma de los montos por el tipo de cambio de la moneda del fondo a la moneda de la cuenta, de todas las inversiones que están asociadas a dicho portafolio, más el valor de Cash Account del portafolio.
	Portafolio.fechaEvaluacionPortafolio = «Inversion.fechaEvaluacion WHERE inversión.idPortafolio = Portafolio.idPortafolio»	La fecha de Evaluacion de cada inversión debe ser igual a la fecha de evaluación del portafolio al cual pertenece.
Cuenta	Ejemplo: Titulares de la cuenta: Juan López y Ana Martínez. Nombre de la cuenta: “Juan López / Ana Martínez”	El nombre de la cuenta es igual a la concatenación de los nombres y apellidos (separados por /) de las personas que son dependientes de la cuenta con rol TITULAR.

Resultado - Algoritmo onto2rdb

Se presenta en la Figura 68 la base de datos obtenida a través del algoritmo Onto2bdr.

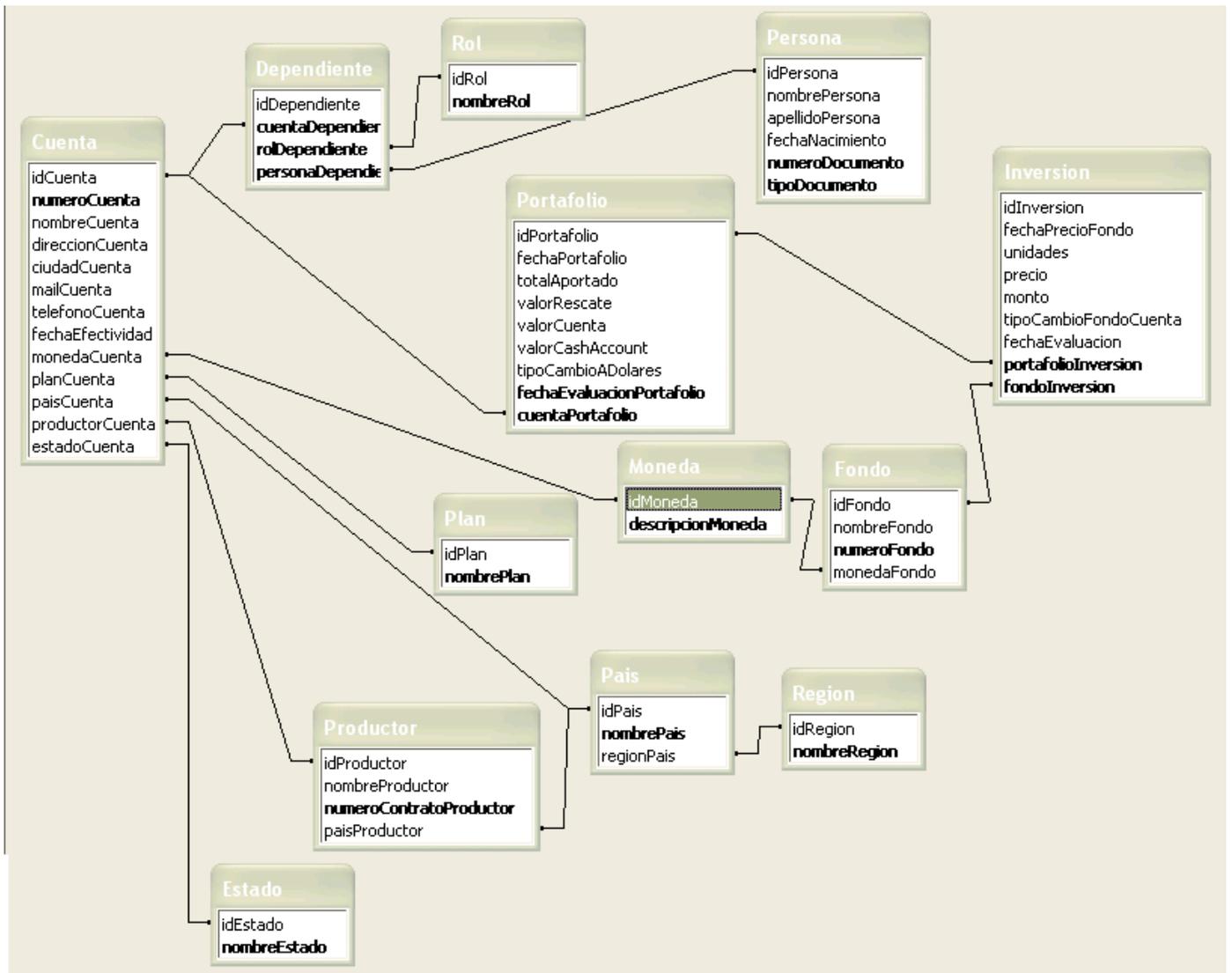


Figura 68. ODS – Resultado de ONTO2RDB

Fase 5 - Generación del diseño lógico y la carga de datos (Algoritmo propuesto por Larrañaga)

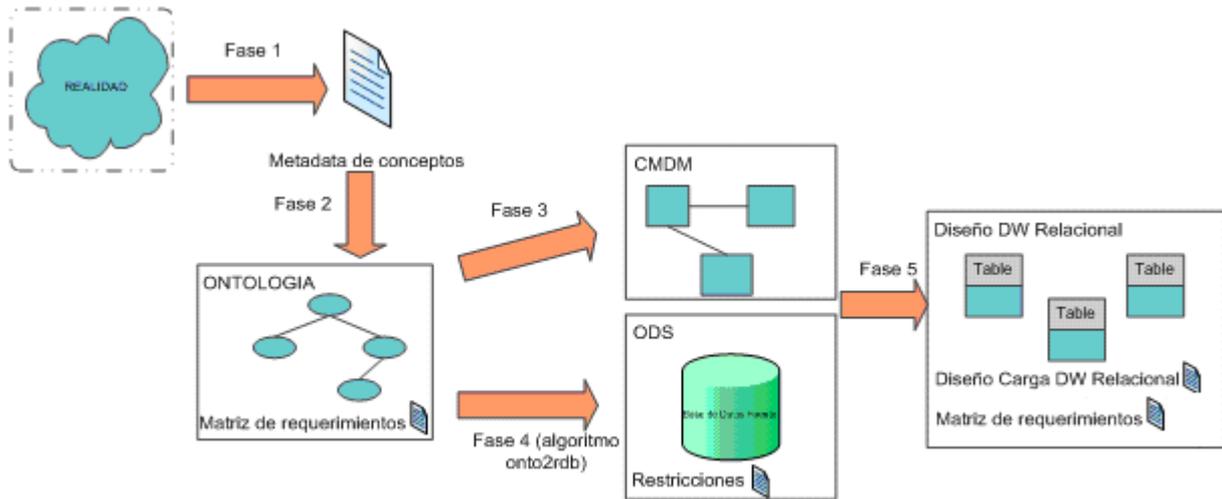


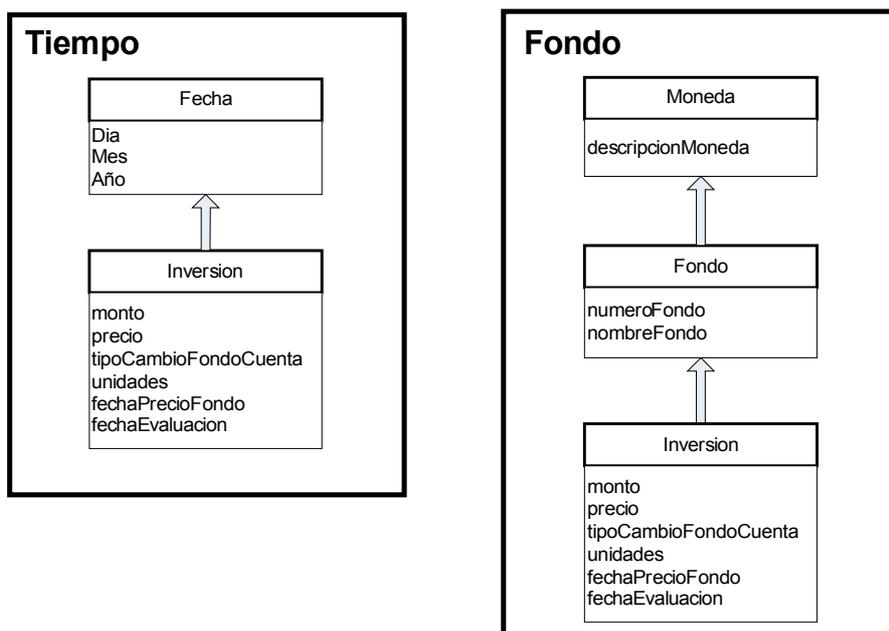
Figura 69. Fase 5

Se utilizarán los resultados obtenidos al ejecutar MD4DW para mostrar que es posible aplicar Larrañaga sobre ellos.

En los primeros cuatro pasos se define todo lo necesario para aplicar el algoritmo en el paso E.5.

E.1 Esquema conceptual

Siguiendo el Caso de Estudio anterior, en el mismo plantea 3 dimensiones: Fondo, Portafolio y Tiempo.



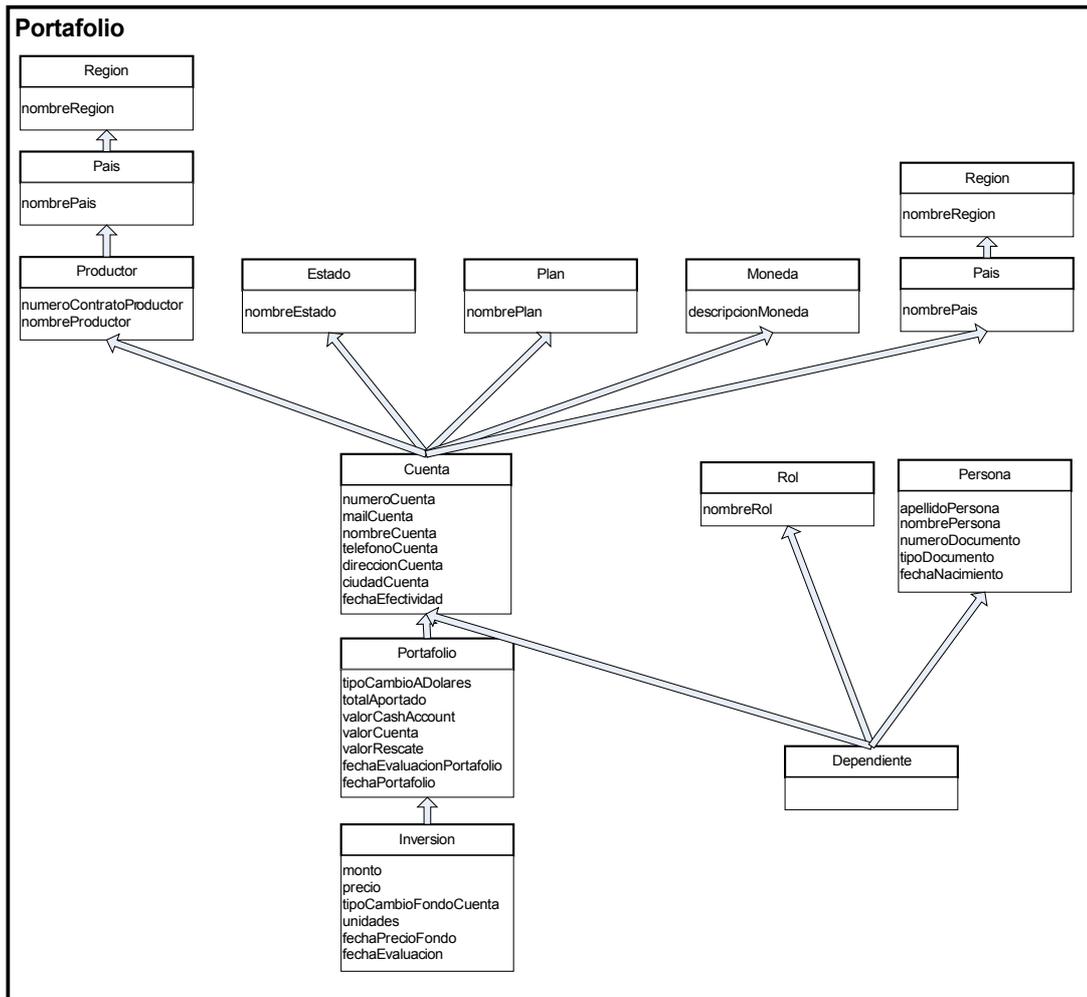


Figura 70. Dimensiones y niveles originales del caso de estudio

Como plantea V. Peralta en su tesis [Per01], es necesario definir una clave por cada nivel y que los nombres de cada nivel sean únicos en el esquema. Siguiendo el mismo criterio que en el paso 2 del algoritmo Onto2rdb ([Paso 2](#)), se definen claves sustitutas para cada nivel, facilitando así el mapeo que debe hacerse más adelante entre el modelo CMDM y la base de datos fuente. Además, se renombran algunos ítems de niveles para mantener la unicidad de los mismos.

En Figura 71 y Figura 72 se presenta el modelo resultante.

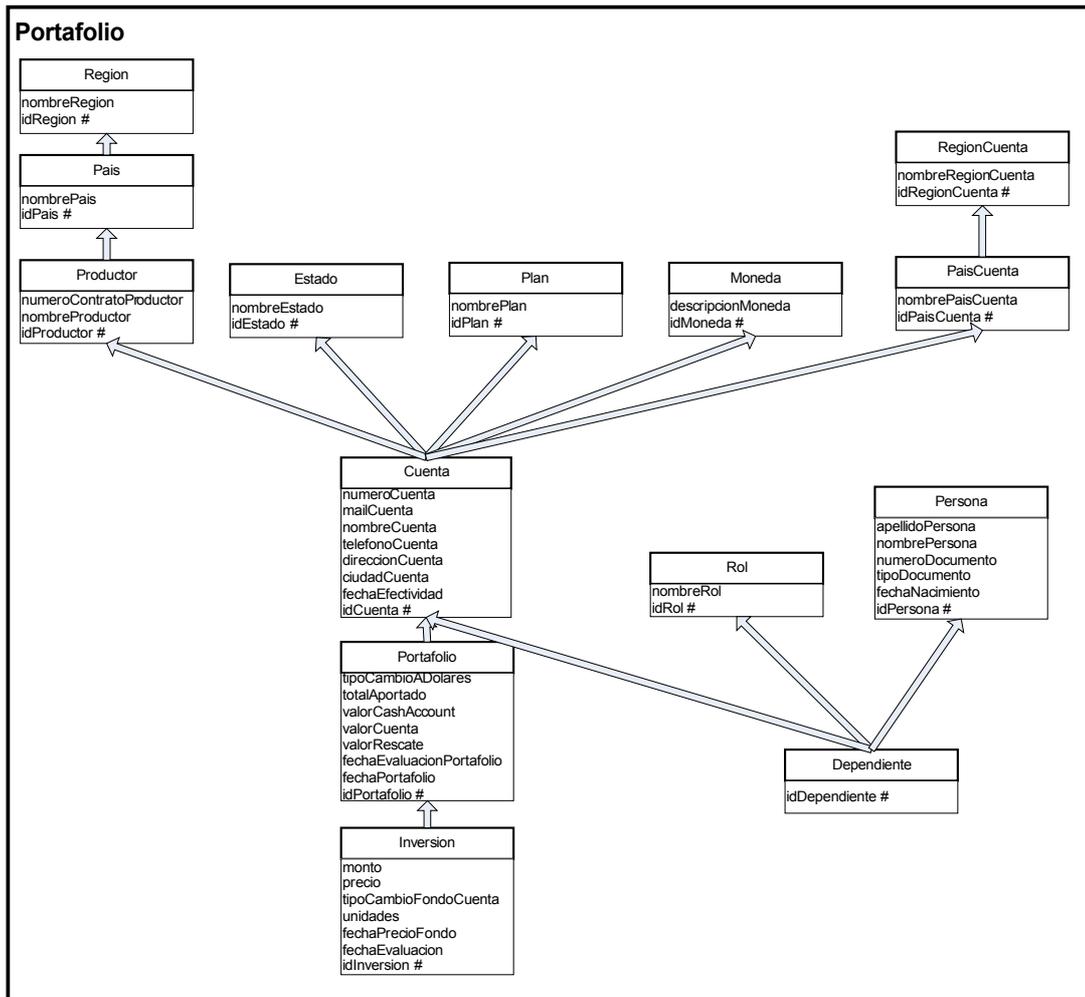


Figura 71. Dimensión Portafolio con claves

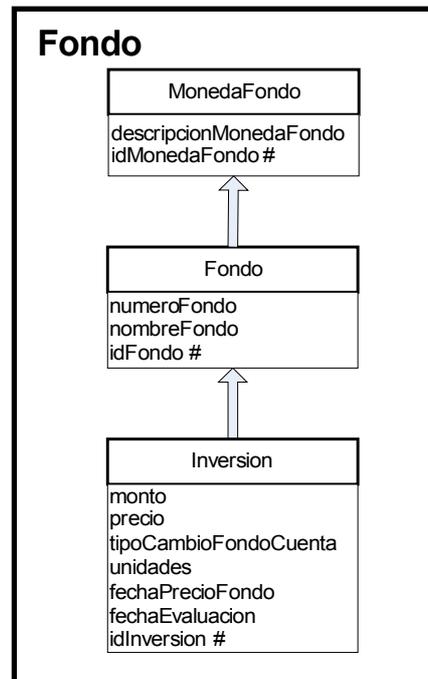
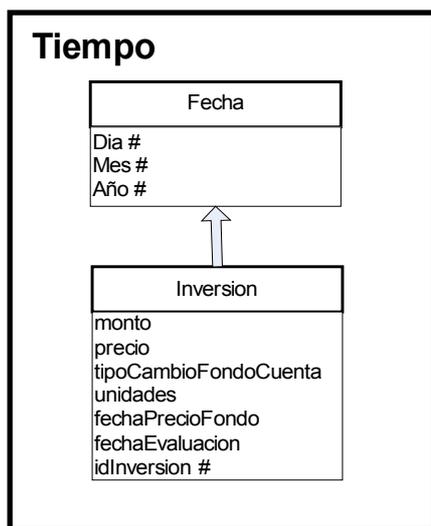


Figura 72. Dimensiones Tiempo y Fondo con clave del caso de estudio

Como se mostró anteriormente, en este caso de estudio se identificó una única relación dimensional.

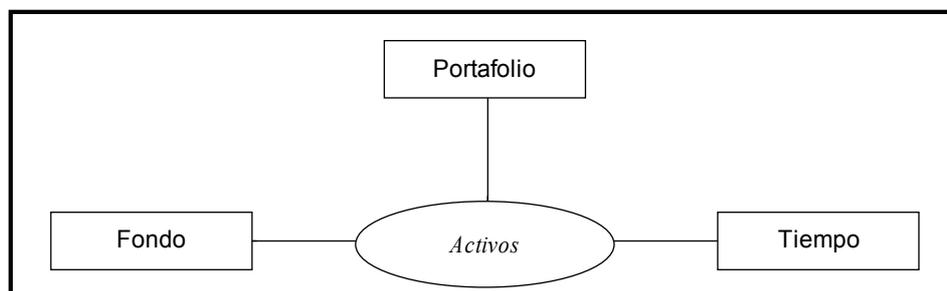


Figura 73. Relación dimensional del caso de estudio

E.2 Base de datos fuente

Descripción del ODS, base en la cual se aplicará el algoritmo de Larrañaga.

Cuenta (idCuenta, numeroCuenta, nombreCuenta, direccionCuenta, ciudadCuenta, mailCuenta, telefonoCuenta, fechaEfectividad, monedaCuenta, planCuenta, paisCuenta, productorCuenta, estadoCuenta)

Refleja una cuenta que tiene inversión en determinados fondos.

Dependiente (idDependiente, cuentaDependiente, rolDependiente, personaDependiente)

Refleja los dependientes de las cuentas, que son personas con un determinado rol en la cuenta.

Rol (idRol, nombreRol)

Representa el rol a través del cual una persona está relacionada con una cuenta.

Estado (idEstado, nombreEstado)

Refleja el estado en el que se puede encontrar una cuenta.

Fondo (idFondo, nombreFondo, numeroFondo, monedaFondo)

Refleja los fondos en los que invierten las cuentas.

Inversion (idInversion, fechaPrecioFondo, unidades, precio, monto, tipoCambioFondoCuenta, fechaEvaluacion, portafolioInversion, fondoInversion)

Refleja la composición del Portafolio de una Cuenta, indicando cuánto y en qué fondo invierte la misma en una fecha determinada. Dicho valor es representado por las unidades del fondo que pertenecen a la cuenta, y el precio de cada unidad a la fecha especificada.

Moneda (IdMoneda, descripcionMoneda)

Corresponde a la moneda de un concepto.

Portafolio (idPortafolio, fechaPortafolio, totalAportado, valorRescate, valorCuenta, valorCashAccount, tipoCambioADolares, fechaEvaluacionPortafolio, cuentaPortafolio)

Refleja los valores totales que interesan de una cuenta con respecto a sus inversiones. Dichos valores corresponden a una determinada fecha y se basan en las inversiones de la cuenta a esa fecha.

Plan (idPlan, nombrePlan)

Refleja la categoría a la que pertenece la cuenta. Dicha categoría define las restricciones, compromisos y beneficios de la cuenta establecidos entre el vendedor de la misma (Productor) y su o sus compradores (Dependientes con Rol Titular).

Region (idRegion, nombreRegion)

Representa una región.

Persona (idPersona, apellidoPersona, nombrePersona, fechaNacimiento, numeroDocumento, tipoDocumento)

Refleja las personas que pueden ser dependientes de una o varias cuentas.

Productor (idProductor, nombreProductor, numeroContratoProductor, paisProductor)

Representa quien realiza la venta de una cuenta.

Pais (idPais, nombrePais, regionPais)

Representa un país.

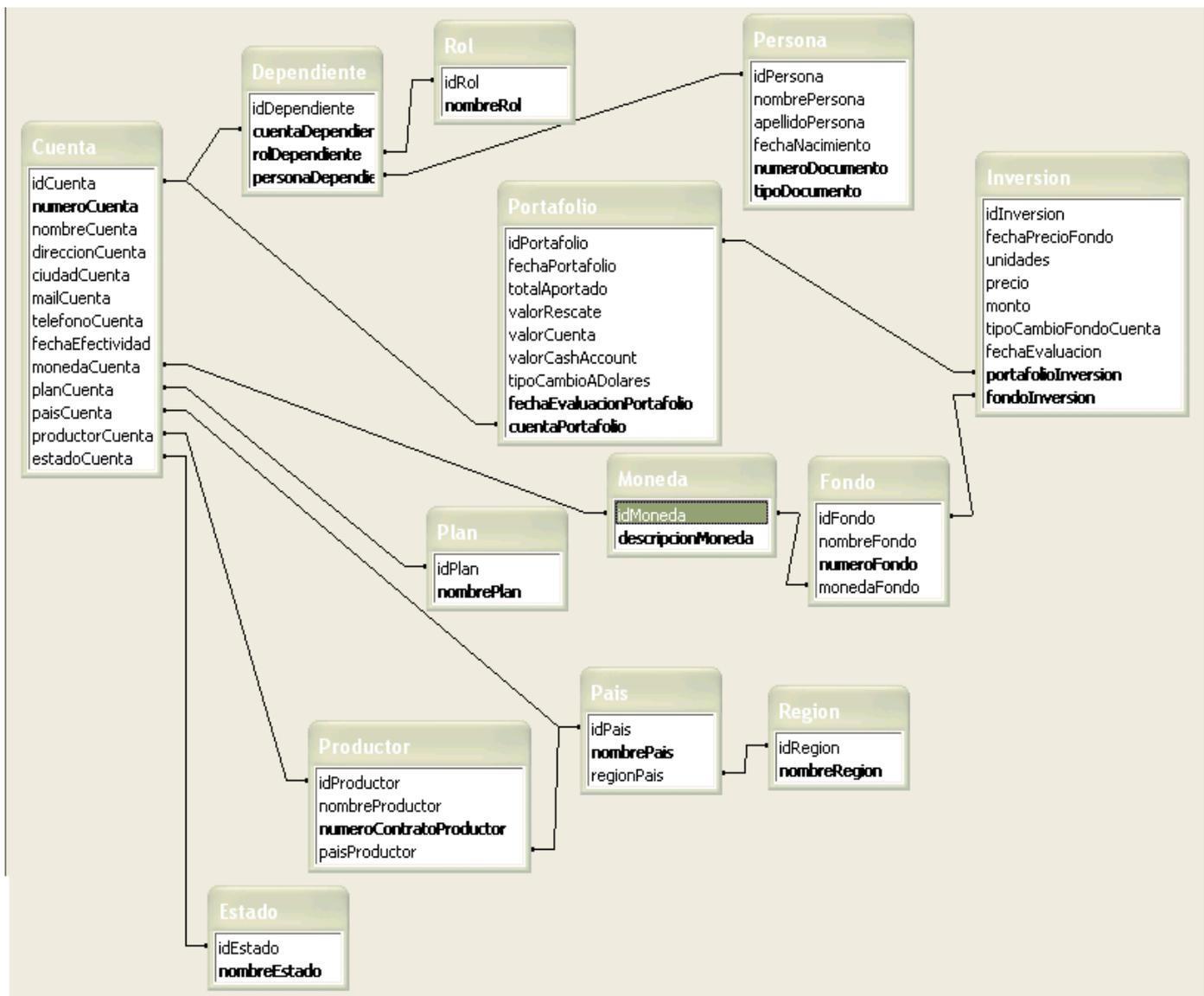


Figura 74. Links utilizados en la BDF

En este caso no es necesario utilizar alias ya que no existen links desde un atributo hacia más de un atributo de otra clase.

E.3 Lineamientos

En base a los requerimientos, se decide materializar tres cubos para la relación dimensional *Activos*:

- 1) Con detalle de Fecha
- 2) Con detalle de Fecha, Fondo
- 3) Con detalle de Fecha, Portafolio

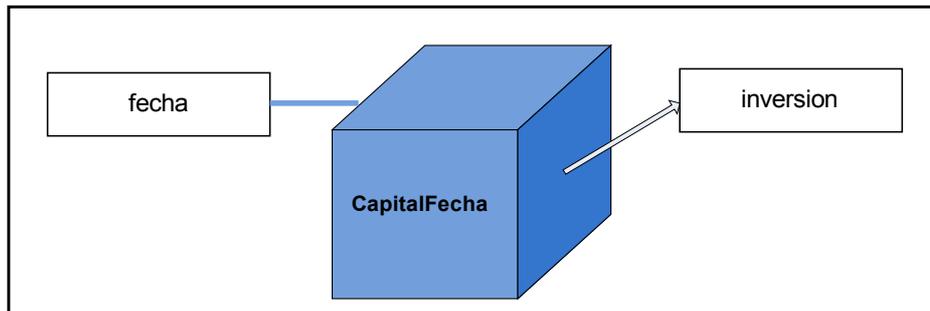


Figura 75. Cubo Capital Fecha

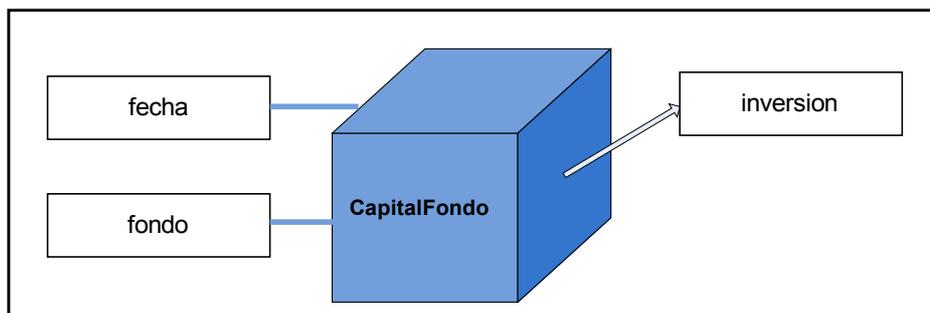


Figura 76. Cubo Capital Fondo

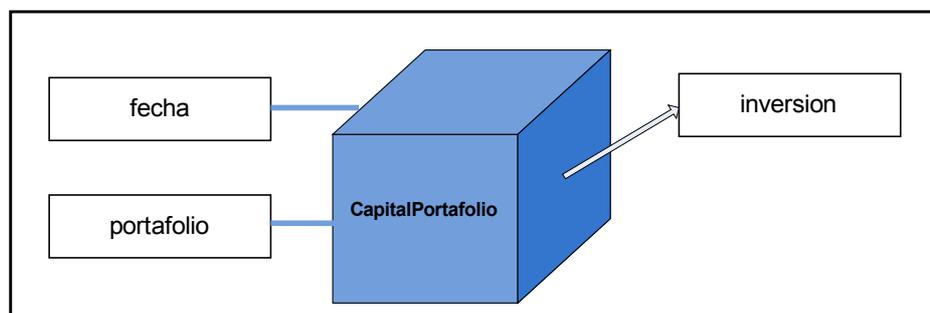


Figura 77. Cubo Capital Portafolio

Con respecto a la fragmentación de cubos, el único cubo para el que se definieron franjas es *CapitalPortafolio*, para el que se definirá una franja que contendrá las fechas correspondientes al año corriente y otra franja para las fechas que pertenecen a años anteriores. Esta fragmentación fue definida en base a las necesidades planteadas por el cliente, dado que se expresó que el 80% de las consultas se realizan sobre información del año corriente. Por lo tanto, para mejorar la performance del cubo se definieron las franjas antes mencionadas.

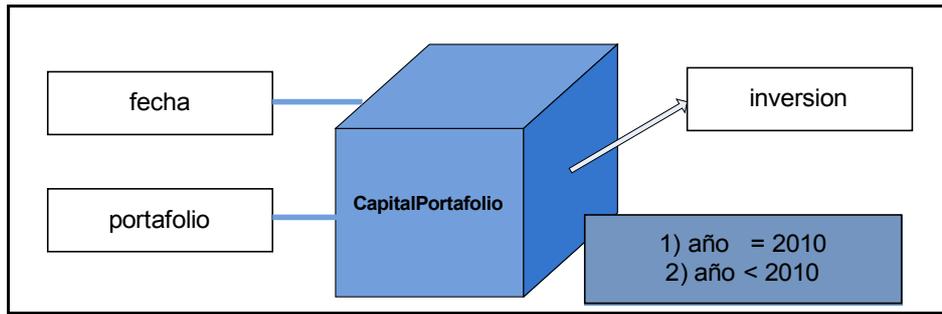


Figura 78. Cubo Capital Portafolio con Fragmentación

Con respecto a la fragmentación de dimensiones, se fragmentó únicamente la dimensión Portafolio, con los fragmentos estado, plan, moneda, portafolio y productor. Cada color representa una fragmentación.

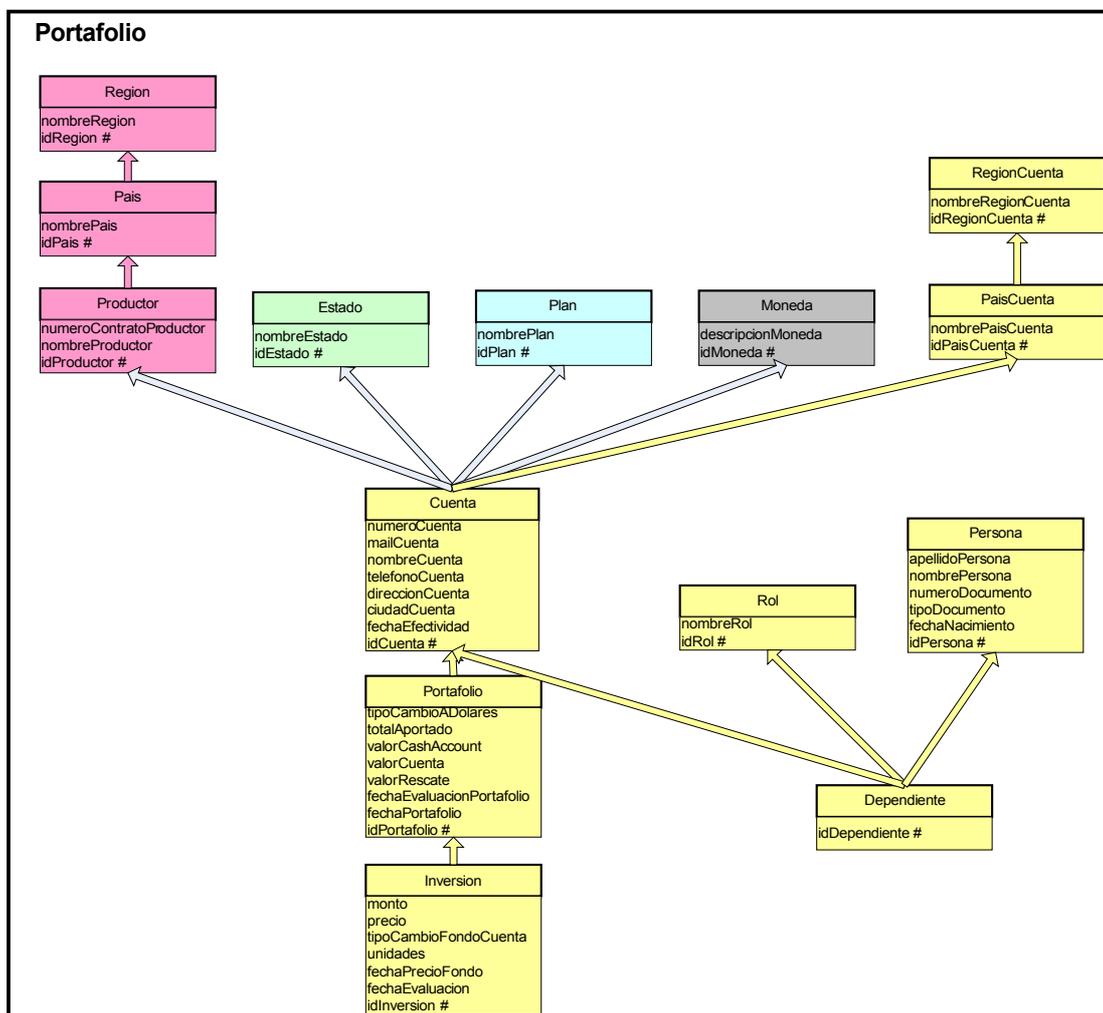


Figura 79. Fragmentación vertical

La fragmentación vertical (Figura 79) fue definida según la relación entre los niveles que componen la dimensión Portafolio. Se agruparon aquellos niveles que suelen requerirse siempre en conjunto y se dejaron independientes los niveles que representan una forma de agrupación para el cliente.

E.4 Correspondencias

Las siguientes figuras (Figura 80., Figura 82, Figura 83, Figura 84 y Figura 85) muestran las correspondencias de los fragmentos de las dimensiones Tiempo, Fondo y Portafolio, y de los cubos CapitalFondo, CapitalPortafolio, CapitalFecha.

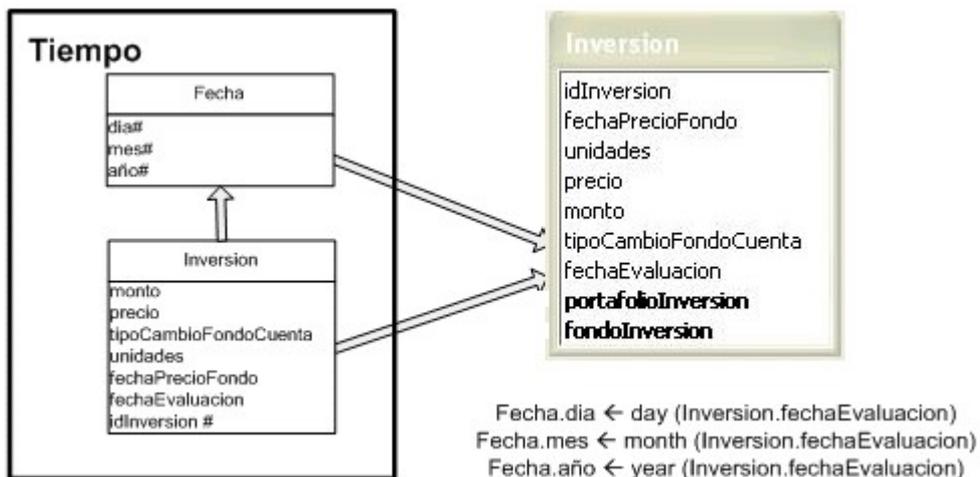


Figura 80. Correspondencias de la dimensión Tiempo

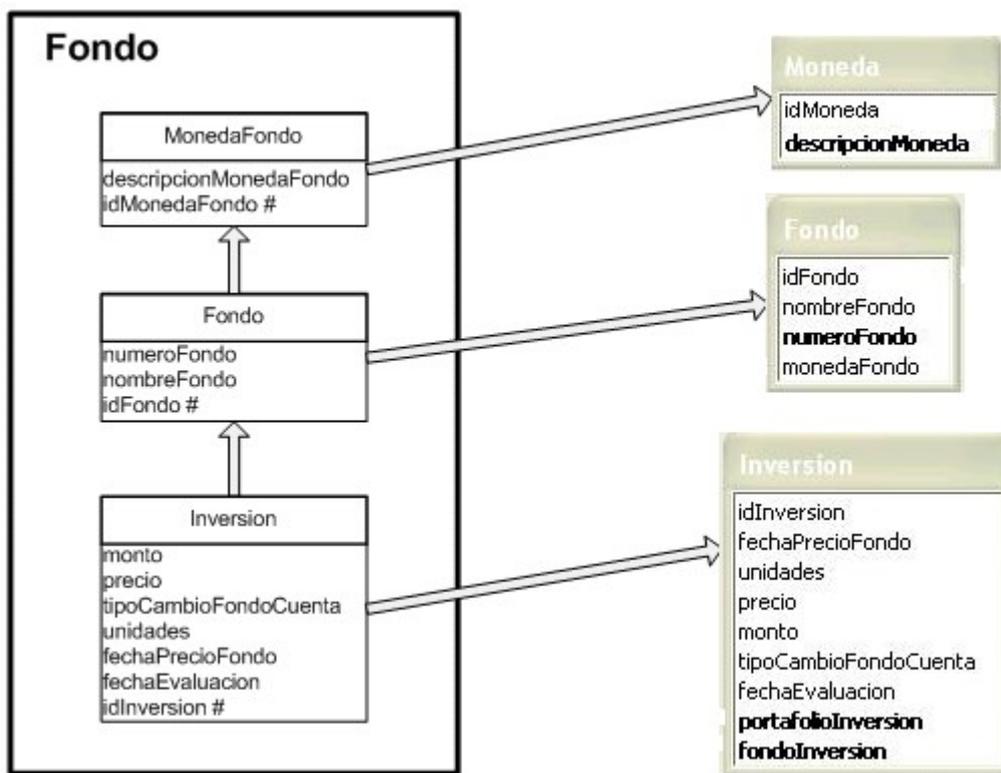


Figura 81. Correspondencias de la dimensión Fondo

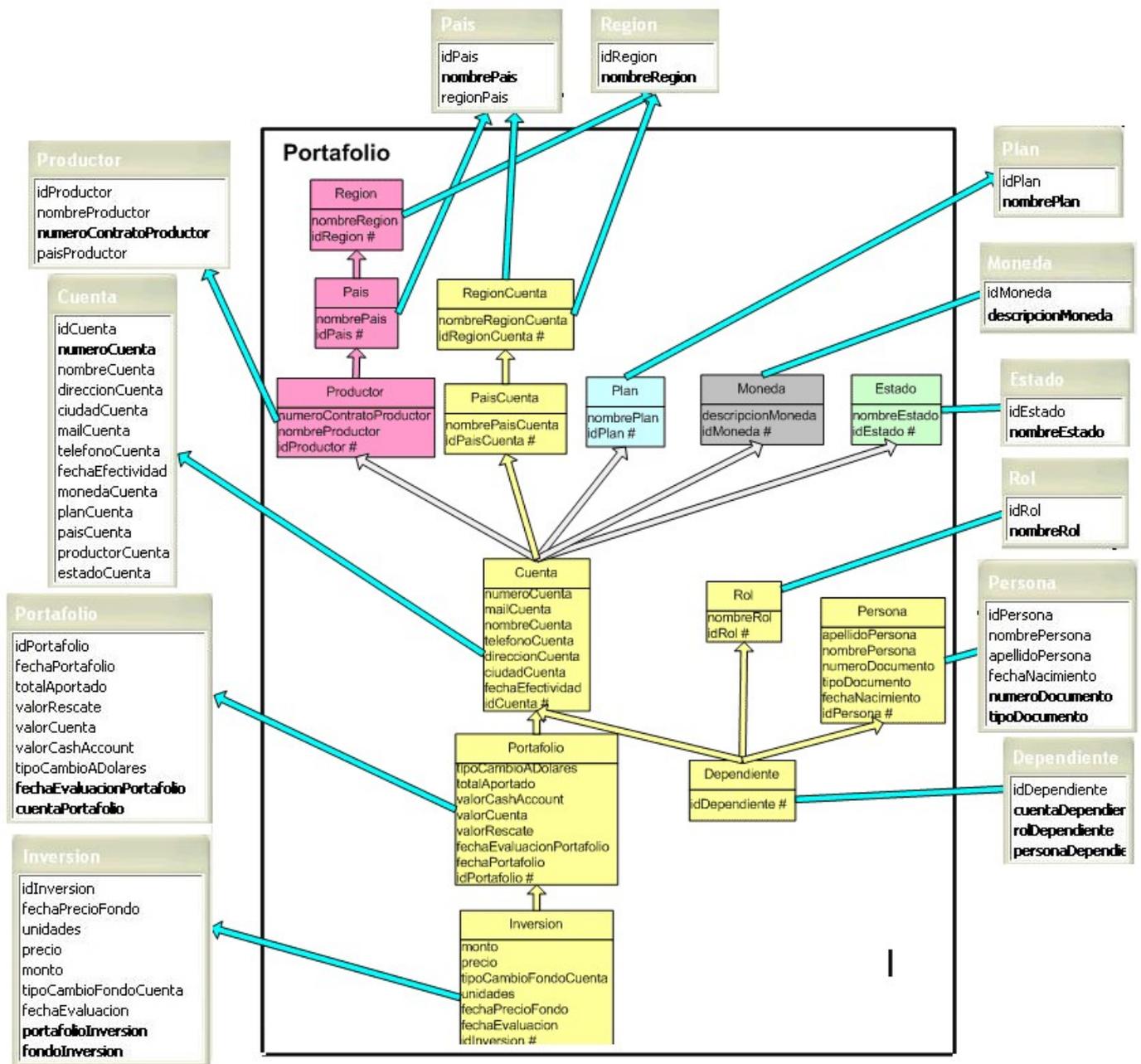


Figura 82. Correspondencias de la dimensión Portafolio

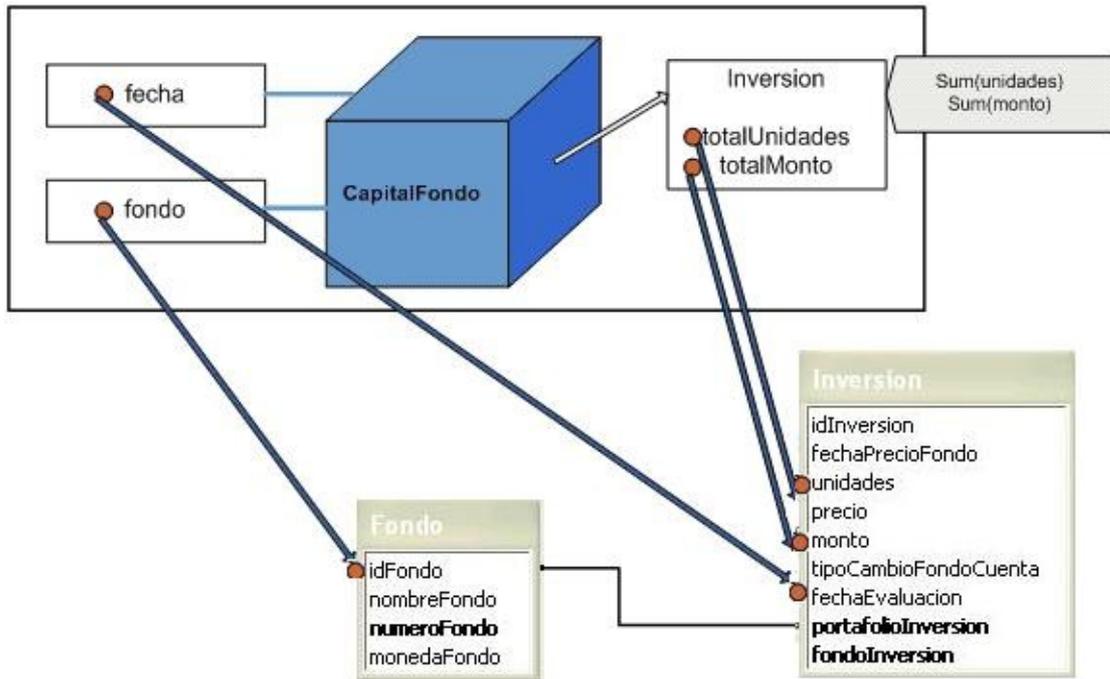


Figura 83. Correspondencias del cubo CapitalFondo

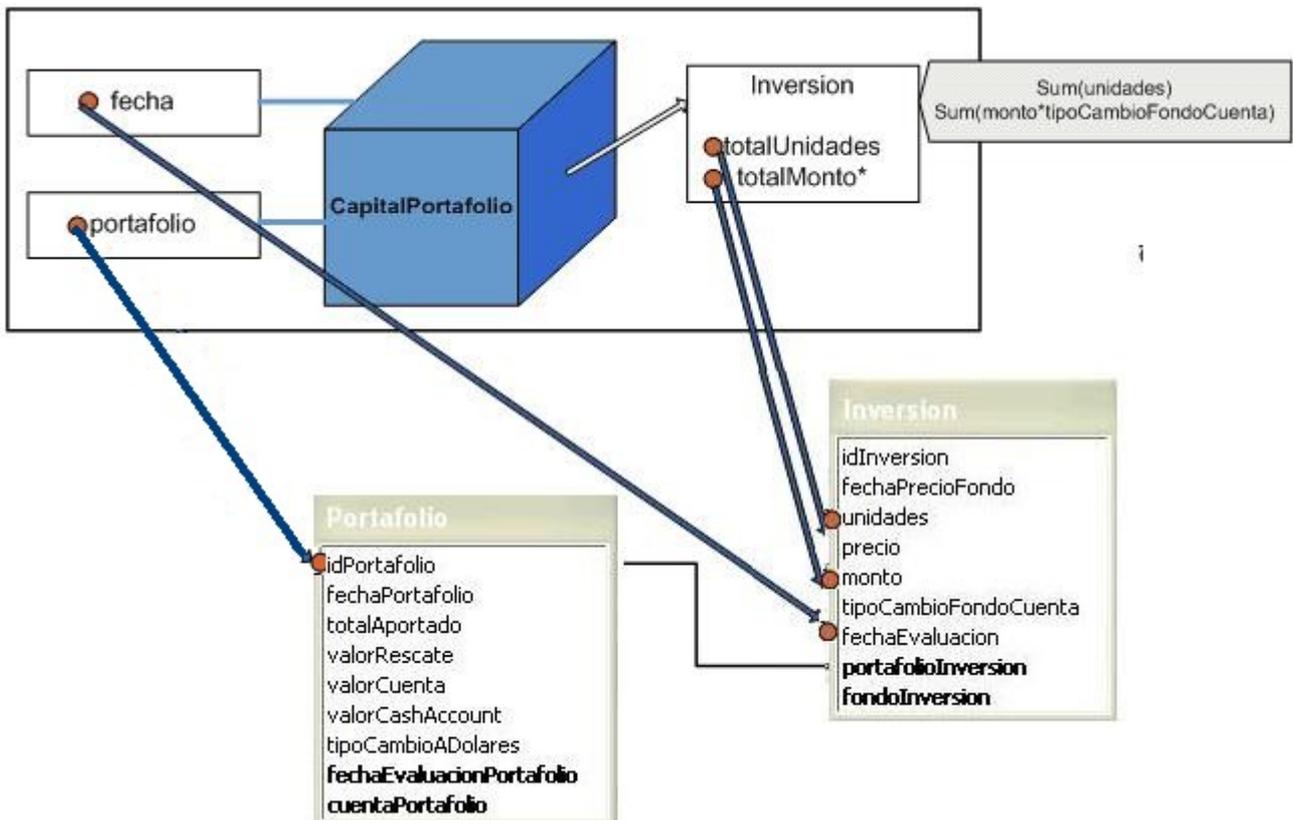


Figura 84. Correspondencias del cubo CapitalPortafolio

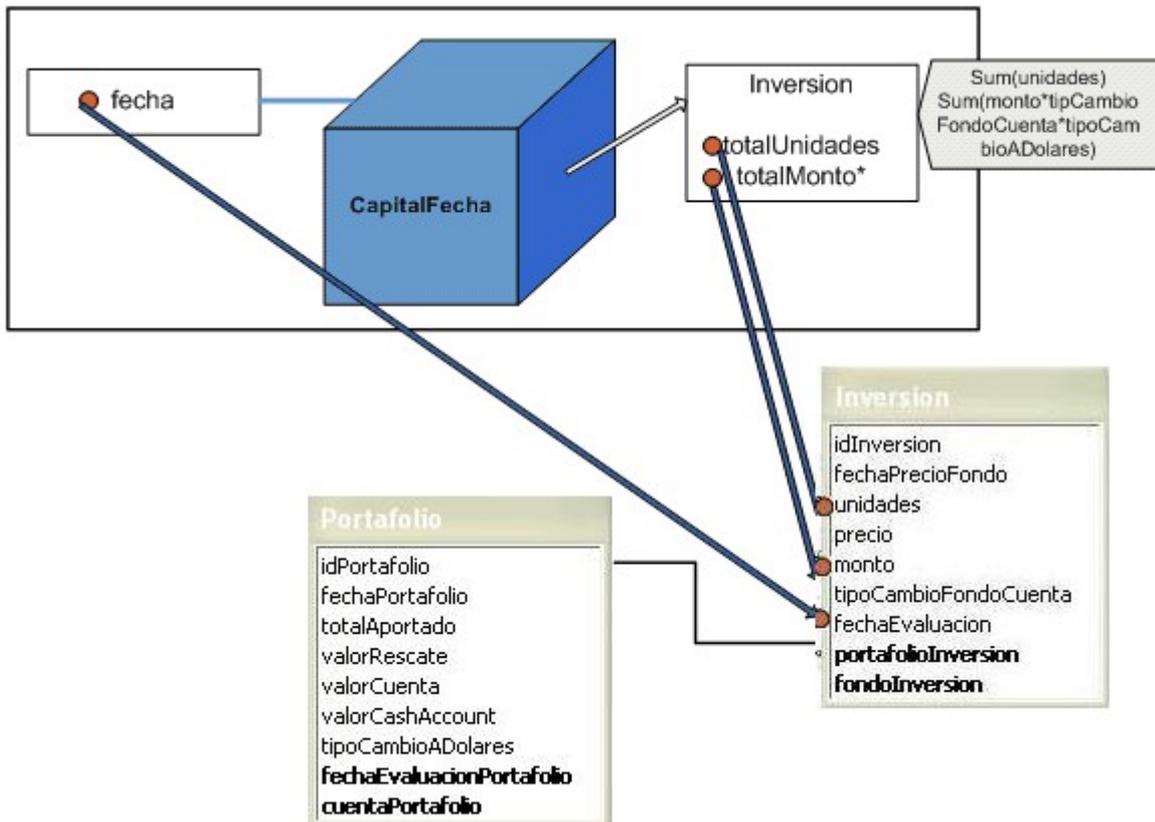


Figura 85. Correspondencias del cubo CapitalFecha

E.5 Aplicación del Algoritmo

- **Tratamiento de dimensiones**

- **Paso 1**

Construir Esqueletos. Tomar cada fragmento y armar una única vista que lo mapee, definiendo así el esqueleto de cada dimensión con los atributos de mapeo directo.

DIMENSION FONDO

```
CREATE VIEW DWFondo
AS
SELECT Fondo.idFondo,Fondo.nombreFondo,Fondo.numeroFondo,
        Fondo.monedaFondo,MonedaFondo.DescripcionMoneda
FROM Fondo,Moneda
WHERE Fondo.monedaFondo= Moneda.idMoneda
```

DIMENSION PORTAFOLIO

Fragmento Dimension Estado

```
CREATE VIEW DWEstado AS
SELECT Estado.idEstado,Estado.nombreEstado
FROM Estado
```

Fragmento Dimension Plan

```
CREATE VIEW DWPlan AS
SELECT Plan.idPlan,Plan.nombrePlan
FROM Plan
```

Fragmento Dimension Moneda

```
CREATE VIEW DWMoneda AS
SELECT Moneda.idMoneda,Moneda.descripcionMoneda
FROM Moneda
```

Fragmento Dimension Productor

```
CREATE VIEW DWProductor AS
SELECT Productor.idProductor,Productor.numeroContratoProductor,
        Productor.nombreProductor,Productor.paisProductor,Pais.nombrePais,
        Pais.regionPais,Region.nombreRegion
FROM Productor,Pais,Region
WHERE Productor.paisProductor = Pais.idPais
AND Pais. regionPais = Region.idRegion
```

Fragmento Dimension Portafolio

```
CREATE VIEW DWPortafolio AS
SELECT Portafolio.idPortafolio,Portafolio.fechaPortafolio,
       Portafolio.fechaEvaluacionPortafolio,Portafolio.valorRescate,
       Portafolio.valorCuenta,Portafolio.valorCashAccount,
       Portafolio.totalaportado,Portafolio.tipoCambioADolares,
       Portafolio.cuentaPortafolio,Cuenta.numeroCuenta,Cuenta.fechaEfectividad,
       Cuenta.ciudadCuenta,Cuenta.direccionCuenta,
       Cuenta.telefonoCuenta,Cuenta.mailCuenta,Cuenta.paisCuenta,
       Pais.nombrePais,Pais.regionPais,Region.nombreRegion,
       Cuenta.productorCuenta,Cuenta.planCuenta,
       Cuenta.monedaCuenta,Cuenta.estadoCuenta
FROM   Portafolio,Cuenta,Pais,Region
WHERE  Portafolio.cuentaPortafolio = Cuenta.idCuenta
       AND Cuenta.paisCuenta = Pais.idPais
       AND Pais.regionPais = Region.idRegion
```

El esquema resultado para los fragmentos de dimensión definidos es el siguiente:

DWFondo (idFondo, nombreFondo, numeroFondo, monedaFondo,
DescripcionMoneda)

DWEstado (idEstado, nombreEstado)

DWPlan (idPlan, nombrePlan)

DWMoneda (idMoneda, descripcionMoneda)

DWProductor (idProductor, numeroContratoProductor, nombreProductor, paisProductor,
nombrePais, regionPais, nombreRegion)

DWPortafolio (idPortafolio, fechaPortafolio, fechaEvaluacionPortafolio,
valorRescate,valorCuenta, valorCashAccount, totalAportado,
tipoCambioADolares, cuentaPortafolio, numeroCuenta, fechaEfectividad,
ciudadCuenta,direccionCuenta, telefonoCuenta, mailCuenta,
paisCuenta, nombrePais, regionPais, nombreRegion, productorCuenta,
planCuenta, monedaCuenta, estadoCuenta)

○ Paso 2

Renombrar los atributos que difieran con el nombre con el que se mapean en CMDM.
Este paso aplica únicamente a los fragmentos de dimensión DWFondo y DWPortafolio.

DIMENSION FONDO

```
CREATE VIEW DWFondo
AS
SELECT Fondo.idFondo,Fondo.nombreFondo,Fondo.numeroFondo,
       Fondo.monedaFondo AS idMonedaFondo,
```

```

        MonedaFondo.descripcionMoneda AS descripcionMonedaFondo
FROM    Fondo,Moneda
WHERE   Fondo.monedaFondo= Moneda.idMoneda

```

DIMENSION PORTAFOLIO

Fragmento Dimension Portafolio

```

CREATE VIEW DWPortafolio AS
SELECT Portafolio.idPortafolio,Portafolio.fechaPortafolio,
        Portafolio.fechaEvaluacionPortafolio, Portafolio.valorRescate,
        Portafolio.valorCuenta,Portafolio.valorCashAccount,
        Portafolio.totalaportado,Portafolio.tipoCambioADolares,
        Portafolio.cuentaPortafolio,Cuenta.numeroCuenta,
        Cuenta.fechaEfectividad,Cuenta.ciudadCuenta,
        Cuenta.direccionCuenta,Cuenta.telefonoCuenta,
        Cuenta.mailCuenta,
        Cuenta.paisCuenta AS idPaisCuenta,
        Pais.nombrePais AS nombrePaisCuenta,
        Pais.regionPais AS idRegionCuenta,
        Region.nombreRegion AS nombreRegionCuenta,
        Cuenta.productorCuenta,Cuenta.planCuenta,Cuenta.monedaCuenta,
        Cuenta.estadoCuenta
FROM    Portafolio,Cuenta,Pais,Region
WHERE   Portafolio.cuentaPortafolio = Cuenta.idCuenta
        AND Cuenta.paisCuenta = Pais.idPais
        AND Pais.regionPais = Region.idRegion

```

Obteniendo el esquema siguiente:

DWFondo (idFondo, nombreFondo, numeroFondo, idMonedaFondo,
DescripcionMonedaFondo)

DWEstado (idEstado, nombreEstado)

DWPlan (idPlan, nombrePlan)

DWMoneda (idMoneda, descripcionMoneda)

DWProductor (idProductor, numeroContratoProductor, nombreProductor, paisProductor,
nombrePais, regionPais, nombreRegion)

DWPortafolio (idPortafolio, fechaPortafolio, fechaEvaluacionPortafolio,
valorRescate, valorCuenta, valorCashAccount, totalAportado,
tipoCambioADolares, cuentaPortafolio, numeroCuenta, fechaEfectividad,
ciudadCuenta, direccionCuenta, telefonoCuenta, mailCuenta,
idPaisCuenta, nombrePaisCuenta, idRegionCuenta,
nombreRegionCuenta, productorCuenta, planCuenta,
monedaCuenta, estadoCuenta)

- **Paso 3**

Generar atributos para ítems con mapeo calculado o externo.

En este paso aparece la dimensión tiempo, ya que sus ítems son calculados sobre el ítem *fecha* del nivel *Inversión*.

DIMENSION TIEMPO

```
CREATE VIEW DWTiempo AS
SELECT day(fechaEvaluacion) AS dia, month(fechaEvaluacion) AS mes,
       year(fechaEvaluacion) AS anio
FROM   Inversion
GROUP BY day(fechaEvaluacion),month(fechaEvaluacion),year(fechaEvaluacion)
```

Esquema resultado:

DWFondo (idFondo, nombreFondo, numeroFondo, idMonedaFondo,
DescripcionMonedaFondo)

DWEstado (idEstado, nombreEstado)

DWPlan (idPlan, nombrePlan)

DWMoneda (idMoneda, descripcionMoneda)

DWProductor (idProductor, numeroContratoProductor, nombreProductor, paisProductor,
nombrePais, regionPais, nombreRegion)

DWPortafolio (idPortafolio, fechaPortafolio, fechaEvaluacionPortafolio,
valorRescate, valorCuenta, valorCashAccount, totalAportado,
tipoCambioADolares, cuentaPortafolio, numeroCuenta, fechaEfectividad,
ciudadCuenta, direccionCuenta, telefonoCuenta, mailCuenta,
idPaisCuenta, nombrePaisCuenta, idRegionCuenta,
nombreRegionCuenta, productorCuenta, planCuenta,
monedaCuenta, estadoCuenta)

DWTiempo (dia,mes,anio)

- **Paso 4**

Este paso no es necesario en el tratamiento de dimensiones.

- **Paso 5**

Este paso no es necesario en el tratamiento de dimensiones.

- **Paso 6**

Ajustar las claves de cada esquema.

El esquema resultado es el siguiente:

DWFondo (idFondo, nombreFondo, numeroFondo, idMonedaFondo, DescripcionMonedaFondo)

DWEstado (idEstado, nombreEstado)

DWPlan (idPlan, nombrePlan)

DWMoneda (idMoneda, descripcionMoneda)

DWProductor (idProductor, numeroContratoProductor, nombreProductor, paisProductor, nombrePais, regionPais, nombreRegion)

DWPortafolio (idPortafolio, fechaPortafolio, fechaEvaluacionPortafolio, valorRescate, valorCuenta, valorCashAccount, totalAportado, tipoCambioADolares, cuentaPortafolio, numeroCuenta, fechaEfectividad, ciudadCuenta, direccionCuenta, telefonoCuenta, mailCuenta, idPaisCuenta, nombrePaisCuenta, idRegionCuenta, nombreRegionCuenta, productorCuenta, planCuenta, monedaCuenta, estadoCuenta)

DWTiempo (dia,mes,anio)

- **Tratamiento de cubos**

- **Paso 7**

Construir Esqueletos. Tomar cada cubo con mapeo directo con la base y armar una única vista que lo mapee, definiendo así el esqueleto de cada cubo con los atributos de mapeo directo.

CUBO CAPITALFECHA

```
CREATE VIEW DWCapitalFecha  
AS
```

```
SELECT Inversion.idInversion, Inversion.fechaPrecioFondo, Inversion.unidades,  
       Inversion.precio, Inversion.monto, Inversion.tipoCambioFondoCuenta,  
       Inversion.fechaEvaluacion, Inversion.portafolioInversion, Inversion.fondoInversion,  
       Portafolio.fechaPortafolio, Portafolio.totalAportado,  
       Portafolio.valorRescate, Portafolio.valorCuenta, Portafolio.valorCashAccount,  
       Portafolio.tipoCambioADolares, Portafolio.fechaEvaluacionPortafolio,  
       Portafolio.cuentaPortafolio
```

```
FROM Inversion, Portafolio
```

```
WHERE Inversion.portafolioInversion = Portafolio.idPortafolio
```

CUBO CAPITALPORTAFOLIO

```

CREATE VIEW DWCapitalPortafolio
AS
SELECT Inversion.idInversion,Inversion.fechaPrecioFondo,Inversion.unidades,
        Inversion.precio,Inversion.monto,Inversion.tipoCambioFondoCuenta,
        Inversion.fechaEvaluacion,Inversion.portafolioInversion,Inversion.fondoInversion,
        Portafolio.fechaPortafolio,Portafolio.totalAportado,
        Portafolio.valorRescate,Portafolio.valorCuenta,Portafolio.valorCashAccount,
        Portafolio.tipoCambioADolares,Portafolio.fechaEvaluacionPortafolio,
        Portafolio.cuentaPortafolio
FROM Inversion,Portafolio
WHERE Inversion.portafolioInversion = Portafolio.idPortafolio

```

CUBO CAPITALFONDO

```

CREATE VIEW DWCapitalFondo
AS
SELECT Inversion.idInversion,Inversion.fechaPrecioFondo,Inversion.unidades,
        Inversion.precio,Inversion.monto,Inversion.tipoCambioFondoCuenta,
        Inversion.fechaEvaluacion,Inversion.portafolioInversion,Inversion.fondoInversion,
        Fondo.nombreFondo,Fondo.numeroFondo,Fondo.monedaFondo
FROM Inversion,Fondo
WHERE Inversion.fondoInversion = Fondo.idFondo

```

El esquema resultado es el siguiente:

DWCapitalFecha (idInversion, fechaPrecioFondo, unidades, precio, monto, tipoCambioFondoCuenta, fechaEvaluacion, portafolioInversion, fondoInversion, fechaPortafolio, totalAportado, valorRescate, valorCuenta, valorCashAccount, tipoCambioADolares, fechaEvaluacionPortafolio, cuentaPortafolio)

DWCapitalPortafolio (idInversion, fechaPrecioFondo, unidades, precio, monto, tipoCambioFondoCuenta, fechaEvaluacion, portafolioInversion, fondoInversion, fechaPortafolio, totalAportado, valorRescate, valorCuenta, valorCashAccount, tipoCambioADolares, fechaEvaluacionPortafolio, cuentaPortafolio)

DWCapitalFondo (idInversion, fechaPrecioFondo, unidades, precio, monto, tipoCambioFondoCuenta, fechaEvaluacion, portafolioInversion, fondoInversion, nombreFondo, numeroFondo, monedaFondo)

- **Paso 8**

No es necesario renombrar ningún atributo ya que ninguno de sus nombres difiere con el nombre con el mapean en CMDM.

- **Paso 9**

Generar atributos para ítems con mapeo calculado o externo. Para los cubos considerados, se deben generar los atributos que corresponden a funciones sobre *fechaEvaluacion* de la tabla *Inversión*, y aplicar la función definida sobre la medida del cubo para aquellos en los que corresponda.

CUBO CAPITALFECHA

```
CREATE VIEW DWCapitalFecha
AS
SELECT Inversion.idInversion, Inversion.fechaPrecioFondo, Inversion.unidades,
       Inversion.precio, Inversion.monto, Inversion.tipoCambioFondoCuenta,
       Inversion.fechaEvaluacion, Inversion.portafolioInversion, Inversion.fondoInversion,
       Portafolio.fechaPortafolio, Portafolio.totalAportado,
       Portafolio.valorRescate, Portafolio.valorCuenta,
       Portafolio.valorCashAccount,
       Portafolio.tipoCambioADolares, Portafolio.fechaEvaluacionPortafolio,
       Portafolio.cuentaPortafolio, day(Inversion.fechaEvaluacion) AS dia,
       month(Inversion.fechaEvaluacion) AS mes,
       year(Inversion.fechaEvaluacion) AS
       anio, (Inversion.monto* Inversion.tipoCambioFondoCuenta*
       Portafolio.tipoCambioADolares) as totalMonto
FROM   Inversion, Portafolio
WHERE  Inversion.portafolioInversion = Portafolio.idPortafolio
```

CUBO CAPITALPORTAFOLIO

```
CREATE VIEW DWCapitalPortafolio
AS
SELECT Inversion.idInversion, Inversion.fechaPrecioFondo, Inversion.unidades,
       Inversion.precio, Inversion.monto, Inversion.tipoCambioFondoCuenta,
       Inversion.fechaEvaluacion, Inversion.portafolioInversion, Inversion.fondoInversion,
       Portafolio.fechaPortafolio, Portafolio.totalAportado,
       Portafolio.valorRescate, Portafolio.valorCuenta,
       Portafolio.valorCashAccount,
       Portafolio.tipoCambioADolares, Portafolio.fechaEvaluacionPortafolio,
       Portafolio.cuentaPortafolio, day(Inversion.fechaEvaluacion) AS dia,
       month(Inversion.fechaEvaluacion) AS mes,
       year(Inversion.fechaEvaluacion) AS anio,
       (Inversion.monto* Inversion.tipoCambioFondoCuenta) AS totalMonto
FROM   Inversion, Portafolio
WHERE  Inversion.portafolioInversion = Portafolio.idPortafolio
```

CUBO CAPITALFONDO

```

CREATE VIEW DWCapitalFondo
AS
SELECT Inversion.idInversion, Inversion.fechaPrecioFondo, Inversion.unidades,
       Inversion.precio, Inversion.monto, Inversion.tipoCambioFondoCuenta,
       Inversion.fechaEvaluacion, Inversion.portafolioInversion, Inversion.fondoInversion,
       Fondo.nombreFondo, Fondo.numeroFondo, Fondo.monedaFondo,
       day(Inversion.fechaEvaluacion) AS dia,
       month(Inversion.fechaEvaluacion) AS mes,
       year(Inversion.fechaEvaluacion) AS anio
FROM   Inversion, Fondo
WHERE  Inversion.fondoInversion = Fondo.idFondo

```

El esquema resultado es el siguiente:

DWCapitalFecha (idInversion, fechaPrecioFondo, unidades, precio, monto, tipoCambioFondoCuenta, fechaEvaluacion, portafolioInversion, fondoInversion, fechaPortafolio, totalAportado, valorRescate, valorCuenta, valorCashAccount, tipoCambioADolares, fechaEvaluacionPortafolio, cuentaPortafolio, dia, mes, anio, totalMonto)

DWCapitalPortafolio (idInversion, fechaPrecioFondo, unidades, precio, monto, tipoCambioFondoCuenta, fechaEvaluacion, portafolioInversion, fondoInversion, fechaPortafolio, totalAportado, valorRescate, valorCuenta, valorCashAccount, tipoCambioADolares, fechaEvaluacionPortafolio, cuentaPortafolio, dia, mes, anio, totalMonto)

DWCapitalFondo (idInversion, fechaPrecioFondo, unidades, precio, monto, tipoCambioFondoCuenta, fechaEvaluacion, portafolioInversion, fondoInversion, nombreFondo, numeroFondo, monedaFondo, dia, mes, anio)

○ **Paso 10**

Se aplican los filtros que definen las franjas horizontales y verticales de los cubos. Según los lineamientos definidos, sólo se tendrá una fragmentación horizontal en el cubo *CapitalPortafolio*.

CUBO CAPITALPORTAFOLIO

```

CREATE VIEW DWCapitalPortafolioActual
AS
SELECT Inversion.idInversion, Inversion.fechaPrecioFondo, Inversion.unidades,
       Inversion.precio, Inversion.monto, Inversion.tipoCambioFondoCuenta,
       Inversion.fechaEvaluacion, Inversion.portafolioInversion, Inversion.fondoInversion,
       Portafolio.fechaPortafolio, Portafolio.totalAportado,
       Portafolio.valorRescate, Portafolio.valorCuenta,
       Portafolio.valorCashAccount, Portafolio.tipoCambioADolares,

```

```

Portafolio.fechaEvaluacionPortafolio,Portafolio.cuentaPortafolio,
day(Inversion.fechaEvaluacion) AS dia,
month(Inversion.fechaEvaluacion) AS mes,
year(Inversion.fechaEvaluacion) AS anio,
(Inversion.monto*Inversion.tipoCambioFondoCuenta) AS totalMonto
FROM Inversion,Portafolio
WHERE Inversion.portafolioInversion = Portafolio.idPortafolio
AND anio = 2010

```

```

CREATE VIEW DWCapitalPortafolioHistorico
AS
SELECT Inversion.idInversion,Inversion.fechaPrecioFondo,Inversion.unidades,
Inversion.precio,Inversion.monto,Inversion.tipoCambioFondoCuenta,
Inversion.fechaEvaluacion,Inversion.portafolioInversion,Inversion.fondolInversion,
Portafolio.fechaPortafolio,Portafolio.totalAportado,
Portafolio.valorRescate,Portafolio.valorCuenta,
Portafolio.valorCashAccount,Portafolio.tipoCambioADolares,
Portafolio.fechaEvaluacionPortafolio,Portafolio.cuentaPortafolio,
day(Inversion.fechaEvaluacion) AS dia,
month(Inversion.fechaEvaluacion) AS mes,
year(Inversion.fechaEvaluacion) AS anio,
(Inversion.monto*Inversion.tipoCambioFondoCuenta) AS totalMonto
FROM Inversion,Portafolio
WHERE Inversion.portafolioInversion = Portafolio.idPortafolio
AND anio <= 2010

```

Se presenta a continuación el nuevo esquema:

DWCapitalFecha (idInversion, fechaPrecioFondo, unidades, precio, monto, tipoCambioFondoCuenta, fechaEvaluacion, portafolioInversion, fondolInversion, fechaPortafolio, totalAportado, valorRescate, valorCuenta, valorCashAccount, tipoCambioADolares, fechaEvaluacionPortafolio, cuentaPortafolio, dia, mes, anio, totalMonto)

DWCapitalPortafolioActual (idInversion, fechaPrecioFondo, unidades, precio, monto, tipoCambioFondoCuenta, fechaEvaluacion, portafolioInversion, fondolInversion, fechaPortafolio, totalAportado, valorRescate, valorCuenta, valorCashAccount, tipoCambioADolares, fechaEvaluacionPortafolio, cuentaPortafolio, dia, mes, anio, totalMonto)

DWCapitalPortafolioHistorico (idInversion, fechaPrecioFondo, unidades, precio, monto, tipoCambioFondoCuenta, fechaEvaluacion, portafolioInversion, fondolInversion, fechaPortafolio, totalAportado, valorRescate, valorCuenta, valorCashAccount, tipoCambioADolares, fechaEvaluacionPortafolio, cuentaPortafolio, dia, mes, anio, totalMonto)

DWCapitalFondo (idInversion, fechaPrecioFondo, unidades, precio, monto, tipoCambioFondoCuenta, fechaEvaluacion, portafolioInversion, fondoInversion, nombreFondo, numeroFondo, monedaFondo, dia, mes, anio)

- **Paso 11**

Deben eliminarse los atributos sin correspondencia y aplicar la función de agregación sobre cada cubo.

CUBO CAPITALFECHA

```
CREATE VIEW DWCapitalFecha
AS
SELECT SUM(Inversion.unidades) AS totalUnidades,
       day(Inversion.fechaEvaluacion) AS dia,
       month(Inversion.fechaEvaluacion) AS mes,
       year(Inversion.fechaEvaluacion) AS anio,
       SUM(Inversion.monto* Inversion.tipoCambioFondoCuenta*
           Portafolio.tipoCambioADolares) as totalMonto
FROM   Inversion,Portafolio
WHERE  Inversion.portafolioInversion = Portafolio.idPortafolio
GROUP BY day(Inversion.fechaEvaluacion),
         month(Inversion.fechaEvaluacion),
         year(Inversion.fechaEvaluacion)
```

CUBO CAPITALPORTAFOLIO

```
CREATE VIEW DWCapitalPortafolioActual
AS
SELECT SUM(Inversion.unidades) AS totalUnidades,
       Inversion.portafolioInversion,
       day(Inversion.fechaEvaluacion) AS dia,
       month(Inversion.fechaEvaluacion) AS mes,
       year(Inversion.fechaEvaluacion) AS anio,
       SUM(Inversion.monto*Inversion.tipoCambioFondoCuenta) AS totalMonto
FROM   Inversion,Portafolio
WHERE  Inversion.portafolioInversion = Portafolio.idPortafolio AND anio = 2010
GROUP BY Inversion.portafolioInversion,
         day(Inversion.fechaEvaluacion),
         month(Inversion.fechaEvaluacion),
         year(Inversion.fechaEvaluacion)
```

```
CREATE VIEW DWCapitalPortafolioHistorico
AS
SELECT SUM(Inversion.unidades) AS totalUnidades,
       Inversion.portafolioInversion,
```

```

    day(Inversion.fechaEvaluacion) AS dia,
    month(Inversion.fechaEvaluacion) AS mes,
    year(Inversion.fechaEvaluacion) AS anio,
    SUM(Inversion.monto*Inversion.tipoCambioFondoCuenta) AS totalMonto
FROM Inversion, Portafolio
WHERE Inversion.portafolioInversion = Portafolio.idPortafolio AND anio <= 2010
GROUP BY Inversion.portafolioInversion,
         day(Inversion.fechaEvaluacion),
         month(Inversion.fechaEvaluacion),
         year(Inversion.fechaEvaluacion)

```

CUBO CAPITALFONDO

```

CREATE VIEW DWCapitalFondo
AS
SELECT SUM(Inversion.unidades) AS totalUnidades,
       SUM(Inversion.monto) AS totalMonto,
       Inversion.fondoInversion,
       day(Inversion.fechaEvaluacion) AS dia,
       month(Inversion.fechaEvaluacion) AS mes,
       year(Inversion.fechaEvaluacion) AS anio
FROM Inversion,Fondo
WHERE Inversion.fondoInversion = Fondo.idFondo
GROUP BY Inversion.fondoInversion,
         day(Inversion.fechaEvaluacion),
         month(Inversion.fechaEvaluacion),
         year(Inversion.fechaEvaluacion)

```

El nuevo esquema obtenido es el siguiente:

DWCapitalFecha (totalUnidades, dia, mes, anio, totalMonto)

DWCapitalPortafolioActual (totalUnidades, portafolioInversion, dia, mes, anio, totalMonto)

DWCapitalPortafolioHistorico (totalUnidades, portafolioInversion, dia, mes, anio, totalMonto)

DWCapitalFondo (totalUnidades, totalMonto, fondoInversion, dia, mes, anio)

- **Paso 12**

No es necesario ajustar ninguna clave sobre el esquema obtenido anteriormente, por lo cual se mantiene el mismo esquema resultado:

DWCapitalFecha (totalUnidades, dia, mes, anio, totalMonto)

DWCapitalPortafolioActual (totalUnidades, portafolioInversion, dia, mes, anio, totalMonto)

DWCapitalPortafolioHistorico (totalUnidades, portafolioInversion, dia, mes, anio, totalMonto)

DWCapitalFondo (totalUnidades, totalMonto, fondoInversion, dia, mes, anio)

- **Pasos Restantes**

Dado que este caso de estudio no cuenta con cubos recursivos, no serán aplicados los pasos restantes definidos en el algoritmo de Peralta.

- **Esquema Lógico Resultado**

La presenta el esquema lógico resultado para el caso de estudio de este trabajo.

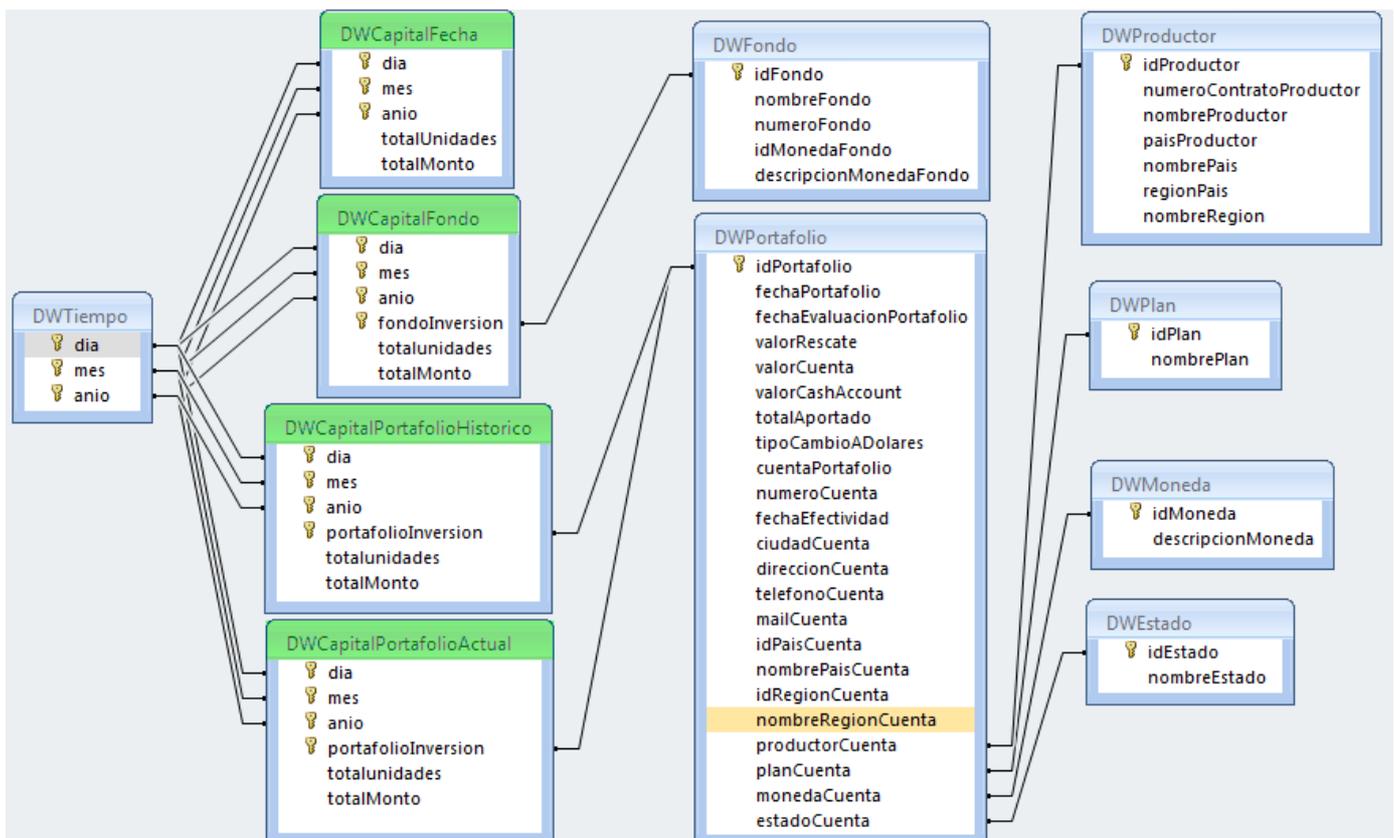


Figura 86. Esquema Lógico Resultado

- **Matriz de trazabilidad completa**

Concepto	RQ1	RQ2	Cubos
Cuenta		√	DWCapitalPortafolioActual DWCapitalPortafolioHistorico DWCapitalFondo DWCapitalFecha
Dependiente		√	
Estado		√	DWCapitalPortafolioActual DWCapitalPortafolioHistorico DWCapitalFondo DWCapitalFecha
Fondo	√	√	DWCapitalPortafolioActual DWCapitalPortafolioHistorico DWCapitalFondo DWCapitalFecha
Inversion	√	√	DWCapitalPortafolioActual DWCapitalPortafolioHistorico DWCapitalFondo DWCapitalFecha
Moneda	√	√	DWCapitalPortafolioActual DWCapitalPortafolioHistorico DWCapitalFondo DWCapitalFecha
Pais		√	DWCapitalPortafolioActual DWCapitalPortafolioHistorico DWCapitalFondo DWCapitalFecha
Persona		√	
Plan		√	DWCapitalPortafolioActual DWCapitalPortafolioHistorico DWCapitalFondo DWCapitalFecha
Portafolio	√	√	DWCapitalPortafolioActual DWCapitalPortafolioHistorico DWCapitalFondo DWCapitalFecha
Productor		√	DWCapitalPortafolioActual DWCapitalPortafolioHistorico DWCapitalFondo DWCapitalFecha
Region		√	DWCapitalPortafolioActual DWCapitalPortafolioHistorico DWCapitalFondo DWCapitalFecha
Rol		√	

Comparación de resultados

En la empresa cliente, [AIVA], hasta el momento se ha trabajado con Datawarehouse y Cubos, pero sin un proceso de desarrollo definido.

Si bien trabajan con herramientas de análisis OLAP, a través de las cuales mantienen cinco cubos, el equipo que hoy en día se encuentra trabajando en dicha empresa únicamente realiza un mantenimiento de los mismos, ya que quienes desarrollaron dichos cubos ya no se encuentran trabajando en la empresa.

No existe documentación de los cubos existentes por lo cual, frente a cualquier cambio en los mismos, un integrante del equipo debe repasar el código e ir “probando” dónde y cómo se debe realizar el cambio.

Desarrollar nuevos cubos es un gran desafío, porque la realidad no está documentada y no existe un proceso de desarrollo definido ni guías para el mismo.

Mientras se desarrollaba este caso de estudio, en AIVA [AIVA] se realizó el mismo proyecto por parte de los integrantes de IT, sin contar con el apoyo de la guía aquí presentada.

Se pueden identificar ventajas del equipo de IT sobre el equipo de este proyecto a la hora de iniciar este caso de estudio, como ser:

- Conocimiento del negocio: mayor entendimiento de los conceptos que participan en la realidad planteada y de los requerimientos sobre la misma.
- Interacción constante con el cliente.

• Equipo Aiva

La forma en que fue desarrollado el cubo requerido por parte de este equipo no siguió una metodología en particular, simplemente fue desarrollado mediante ensayo y error, enfrentando nuevas herramientas y viendo cómo respondían ante los requerimientos planteados.

Se pudo constatar que la fase de análisis fue mucho más corta por parte del equipo de AIVA [AIVA] comparado con el tiempo de análisis empleado por el equipo de este proyecto. Esto se debió a que el equipo de AIVA [AIVA] ya conocía el negocio y los conceptos que en él se manejan. Pero el este equipo no dejó ninguna documentación del análisis realizado.

Con respecto al diseño del DW, el equipo de AIVA [AIVA] no realizó un diseño específico del datawarehouse que estaba desarrollando. Se basó en el poco conocimiento de Datawarehouse que tenía, adaptando a medida el esquema diseñado a las necesidades planteadas. En esta fase, éste equipo tuvo varias vueltas hacia atrás, dado que se iba realizando la carga a medida que se iba desarrollando el Datawarehouse. Frente a cada cambio del esquema lógico del DW, fue requerido adaptar la carga a esos cambios, implicando tiempos extras.

• Equipo de este proyecto

En la etapa de diseño y aplicando la guía propuesta, el equipo de este proyecto alcanzó un resultado similar al desarrollado por el equipo de AIVA [AIVA] en un tiempo menor.

La aplicación de la guía permitió paralelizar las etapas de preparación del ODS con la etapa de transformación del modelo conceptual en el modelo multidimensional del DW en CMDM.

• Análisis del resultado

Analizando el resultado de la aplicación de la guía, se detectó que:

- Es sumamente importante mantener documentación del análisis realizado sobre los requerimientos, tanto para mantener por escrito dicho conocimiento, así como para validar ante el cliente o bien para integrar nuevos participantes en el proyecto. El equipo de AIVA [AIVA] (aunque ya conociera el negocio y el cliente), se enfrentó a varias ambigüedades con respecto a la información que se iba a manejar. La guía propone documentar toda la realidad mediante la Metadata de conceptos y la ontología.
- Aplicando la guía propuesta por este proyecto, se realizaron menos iteraciones durante el diseño en comparación con el equipo de Aiva. Consultado éste último, estas iteraciones surgieron por errores al identificar dimensiones, jerarquías y medidas desde el inicio del diseño. Esta guía da soporte a la identificación de los conceptos de dimensión, jerarquía y medida en forma temprana, ordenada y considerando toda la realidad. Esto se logra a través de las reglas expresadas en la fase 3 de la misma.
- Con respecto a la carga del DW el equipo de AIVA [AIVA] tuvo muchas iteraciones ya que la misma es dependiente del diseño y al tener cambios en este, se tenía que cambiar la carga. La guía, una vez definido el ODS, permite paralelizar el diseño del DW y la carga del ODS. Nótese que el ODS no va a sufrir más cambios (surge del modelo de la realidad relevado y validado), por lo que no debería cambiarse la carga de la misma a menos que se modifiquen las fuentes de datos (que no depende del desarrollo del DW).

El cuadro siguiente presenta los tiempos de desarrollo de cada equipo. Dichos valores fueron tomados de la planilla de horas en el caso del equipo de este proyecto (administrada por el equipo durante todas las etapas del proyecto) y del reporte de horas que el cliente (AIVA) envió.

Equipo de proyecto	Horas	AIVA	Horas
Fase 1	60	Análisis	40
Fase 2	40		
Fase 3	20	Diseño e Implementación	160
Fase 4	10		
Fase 5	30		

Conclusiones del caso de estudio

A través del desarrollo de este caso de estudio, se observó que la guía propuesta resultó exitosa para dicho caso, tanto en la construcción del diseño de un Datawarehouse como en el diseño de la traza de su carga.

Además, desde las primeras fases de la guía, orienta a la generación de documentación que será fundamental para comprender y mantener el datawarehouse en el futuro.

Un punto a destacar es que el desarrollo del datawarehouse es independiente de la(s) fuente(s) de datos de la(s) que se extraerán los datos. Analizando costo/beneficio de lo que propone esta guía sobre este punto, surge el costo extra del desarrollo del mapeo de la base de datos de producción al ODS así como de las restricciones a implementar definidas al final de la aplicación del algoritmo Onto2rdb. Pero a su vez la independencia del datawarehouse de las fuentes permite poder modificar el esquema de la base de producción, la base en su totalidad o integrar varias fuentes, sin que esto afecte el datawarehouse. El impacto de dicha modificación estará concentrado en la función de mapeo de las fuentes al ODS. También debe considerarse el caso de un proyecto en el cual se desarrolle un sistema en su totalidad: con la guía propuesta, los desarrollos de las fuentes y del datawarehouse podrían realizarse en paralelo, optimizando así los tiempos del proyecto.

Metadata de conceptos

Nombre:	Cuenta	Fecha:	14/11/2009
Descripción:	<i>Refleja una cuenta que tiene inversión en determinados fondos</i>		
Responsable:	<i>Nathalie Deppen, Paola Ricca, Diego Trías</i>		
Fuente:			
Usos:	<i>Representa el concepto cuenta dentro del sistema</i>		
Supuestos:			
Identificador:	<i>Número</i>		
Atributos:			
Nombre:	<i>Numero</i>		
Descripción:	<i>Número de la cuenta</i>		
Obligatoriedad:	<i>Este dato es obligatorio</i>		
Concepto Rel. :			
Nombre:	<i>Nombre</i>		
Descripción:	<i>Es la concatenación de los nombres y apellidos de los titulares de la cuenta</i>		
Obligatoriedad:	<i>Este dato es obligatorio</i>		
Concepto Rel. :			
Nombre:	<i>Moneda</i>		
Descripción:	<i>Moneda de la cuenta</i>		
Obligatoriedad:	<i>Este dato es obligatorio</i>		
Concepto Rel. :	<i>Moneda</i>		
Nombre:	<i>Plan</i>		
Descripción:	<i>Plan de la cuenta</i>		
Obligatoriedad:	<i>Este dato es obligatorio</i>		
Concepto Rel. :	<i>Plan</i>		
Nombre:	<i>Estado</i>		
Descripción:	<i>Estado de la cuenta</i>		
Obligatoriedad:	<i>Este dato es obligatorio</i>		
Concepto Rel. :	<i>Estado</i>		
Nombre:	<i>Dirección</i>		
Descripción:	<i>Dirección de contacto de la cuenta</i>		
Concepto Rel. :			
Nombre:	<i>Ciudad</i>		
Descripción:	<i>Ciudad de contacto de la cuenta</i>		
Concepto Rel. :			
Nombre:	<i>País</i>		
Descripción:	<i>País de la dirección de contacto de la cuenta</i>		
Obligatoriedad:	<i>Este dato es obligatorio</i>		
Concepto Rel. :	<i>País</i>		
Nombre:	<i>Mail</i>		
Descripción:	<i>Mail de contacto de la cuenta</i>		
Concepto Rel. :			

Nombre:	<i>Teléfono</i>
Descripción:	<i>Teléfono de contacto de la cuenta</i>
Concepto Rel. :	
Nombre:	<i>Fecha Efectividad</i>
Descripción:	<i>Fecha a partir de la cual la cuenta tiene inversiones</i>
Obligatoriedad:	<i>Este dato es obligatorio</i>
Concepto Rel. :	
Nombre:	<i>Dependientes</i>
Descripción:	<i>Persona relacionadas a la cuenta a través de un rol</i>
Cardinalidad:	<i>N</i>
Obligatoriedad:	<i>Este dato es obligatorio</i>
Concepto Rel. :	Dependiente
Nombre:	<i>Portafolio</i>
Descripción:	<i>Evaluaciones de las inversiones de la cuenta a una determinada fecha</i>
Cardinalidad:	<i>N</i>
Obligatoriedad:	<i>Este dato es obligatorio</i>
Concepto Rel. :	Portafolio
Nombre:	<i>Productor</i>
Descripción:	<i>Vendedor de la cuenta</i>
Obligatoriedad:	<i>Este dato es obligatorio</i>
Concepto Rel. :	Productor

Nombre:	Dependiente	Fecha:	14/11/2009
Descripción:	<i>Refleja los dependientes de las cuentas, que son personas con un determinado rol en la cuenta</i>		
Responsable:	<i>Nathalie Deppen, Paola Ricca, Diego Trías</i>		
Fuente:			
Usos:	<i>Representa las relaciones entre las personas y las cuentas a través de un rol</i>		
Supuestos:			
Identificador:	<i>[Rol, Persona, Cuenta]</i>		
Atributos:			
Nombre:	<i>Cuenta</i>		
Descripción:	<i>Cuenta a la que pertenece el dependiente</i>		
Obligatoriedad:	<i>Este dato es obligatorio</i>		
Concepto Rel. :	Cuenta		
Nombre:	<i>Rol</i>		
Descripción:	<i>Rol de la persona en la cuenta</i>		
Obligatoriedad:	<i>Este dato es obligatorio</i>		
Concepto Rel. :	Rol		
Nombre:	<i>Persona</i>		
Descripción:	<i>Persona que es dependiente</i>		
Obligatoriedad:	<i>Este dato es obligatorio</i>		
Concepto Rel. :	Persona		

Concepto sólo utilizado en la primera ejecución del caso de estudio:

Nombre:	Documento	Fecha:	14/11/2009
Descripción:	<i>Representa el documento de identidad de una persona</i>		
Responsable:	<i>Nathalie Deppen, Paola Ricca, Diego Trías</i>		
Fuente:			
Usos:	<i>Representa documento que identifica a las personas</i>		
Supuestos:			
Identificador:	<i>[Tipo de documento, Número de documento]</i>		
Atributos:			
Nombre:	<i>Tipo Documento</i>		
Descripción:	<i>Tipo de documento de una persona</i>		
Obligatoriedad:	<i>Este dato es obligatorio</i>		
Concepto Rel. :			
Nombre:	<i>Número Documento</i>		
Descripción:	<i>Número de documento de una persona</i>		
Obligatoriedad:	<i>Este dato es obligatorio</i>		
Concepto Rel. :			

Nombre:	Estado	Fecha:	14/11/2009
Descripción:	<i>Refleja el estado en el que se puede encontrar una cuenta</i>		
Responsable:	<i>Nathalie Deppen, Paola Ricca, Diego Trías</i>		
Fuente:			
Usos:	<i>Refleja el estado de una cuenta</i>		
Supuestos:			
Identificador:	<i>Nombre</i>		
Atributos:			
Nombre:	<i>Nombre</i>		
Descripción:	<i>Nombre del estado</i>		
Obligatoriedad:	<i>Este dato es obligatorio</i>		
Concepto Rel. :			

Nombre:	Fondo	Fecha:	14/11/2009
Descripción:	<i>Refleja los fondos en los que invierten las cuentas</i>		
Responsable:	<i>Nathalie Deppen, Paola Ricca, Diego Trías</i>		
Fuente:			
Usos:	<i>Representa los fondos en los que invierten las cuentas</i>		

Supuestos:	
Identificador:	<i>Número Fondo</i>
Atributos:	
Nombre:	<i>Nombre</i>
Descripción:	<i>Nombre del fondo</i>
Obligatoriedad:	<i>Este dato es obligatorio</i>
Concepto Rel. :	
Nombre:	<i>Moneda</i>
Descripción:	<i>Moneda del fondo</i>
Obligatoriedad:	<i>Este dato es obligatorio</i>
Concepto Rel. :	Moneda
Nombre:	<i>Número Fondo</i>
Descripción:	<i>Número que identifica el fondo</i>
Obligatoriedad:	<i>Este dato es obligatorio</i>
Concepto Rel. :	

Nombre:	Inversión	Fecha:	14/11/2009
Descripción:	<i>Representa los elementos que componen el Portafolio de una Cuenta, indicando cuánto y en qué fondo invierte la misma en una fecha determinada. Dicho valor es representado por las unidades del fondo que pertenecen a la cuenta, y el precio de cada unidad a la fecha especificada.</i>		
Responsable:	<i>Nathalie Deppen, Paola Ricca, Diego Trías</i>		
Fuente:			
Usos:	<i>Representa la inversión medida en unidades y dinero de una cuenta en un fondo</i>		
Supuestos:	<i>El precio de cada unidad es expresado en la moneda del fondo</i>		
Identificador:	<i>[Fondo,Portafolio]</i>		
Atributos:			
Nombre:	<i>Fecha Precio Inversion</i>		
Descripción:	<i>Fecha en la cual se evaluó del precio del fondo.</i>		
Obligatoriedad:	<i>Este dato es obligatorio</i>		
Concepto Rel. :			
Nombre:	<i>Fondo</i>		
Descripción:	<i>Fondo de la inversión</i>		
Obligatoriedad:	<i>Este dato es obligatorio</i>		
Concepto Rel. :	Fondo		
Nombre:	<i>Unidades</i>		
Descripción:	<i>Cantidad de unidades del fondo que pertenecen a la cuenta</i>		
Obligatoriedad:	<i>Este dato es obligatorio</i>		
Concepto Rel. :			
Nombre:	<i>Precio</i>		
Descripción:	<i>Precio de cada unidad del fondo</i>		
Obligatoriedad:	<i>Este dato es obligatorio</i>		
Concepto Rel. :			
Nombre:	<i>Monto</i>		

Descripción:	<i>Valor total de la inversión en el fondo</i>
Obligatoriedad:	<i>Este dato es obligatorio</i>
Dato calculado:	<i>Es el producto de Unidades por Precio</i>
Concepto Rel. :	
Nombre:	<i>Tipo de Cambio</i>
Descripción:	<i>Representa el tipo de cambio para pasar de la moneda del fondo a la moneda de la cuenta</i>
Obligatoriedad:	<i>Este dato es obligatorio</i>
Concepto Rel. :	
Nombre:	<i>Fecha Evaluación</i>
Descripción:	<i>Fecha en la que se está evaluando la inversión. Esta fecha debe coincidir con la fecha del portafolio al cual corresponde la inversión.</i>
Obligatoriedad:	<i>Este campo es obligatorio</i>
Concepto Rel. :	
Nombre:	<i>Portafolio</i>
Descripción:	<i>Portafolio al que pertenece la inversión</i>
Obligatoriedad:	<i>Este campo es obligatorio</i>
Concepto Rel. :	<i>Portafolio</i>

Nombre:	Moneda	Fecha:	14/11/2009
Descripción:	<i>Corresponde a la moneda de un concepto</i>		
Responsable:	<i>Nathalie Deppen, Paola Ricca, Diego Trías</i>		
Fuente:			
Usos:	<i>Refleja la moneda de una cuenta o de un fondo</i>		
Supuestos:			
Identificador:	<i>Descripción</i>		
Atributos:			
Nombre:	<i>Descripción</i>		
Descripción:	<i>Describe la abreviación de la moneda</i>		
Obligatoriedad:	<i>Este dato es obligatorio</i>		
Concepto Rel. :			

Nombre:	País	Fecha:	14/11/2009
Descripción:	<i>Representa un país</i>		
Responsable:	<i>Nathalie Deppen, Paola Ricca, Diego Trías</i>		
Fuente:			
Usos:	<i>Refleja un país de un productor o una cuenta</i>		
Supuestos:	<i>Cada país pertenece a una única región</i>		
Identificador:	<i>Nombre</i>		
Atributos:			
Nombre:	<i>Nombre</i>		
Descripción:	<i>Nombre del país</i>		

Obligatoriedad:	<i>Este dato es obligatorio</i>
Concepto Rel. :	
Nombre:	<i>Región</i>
Descripción:	<i>Región a la que pertenece el país</i>
Obligatoriedad:	<i>Este dato es obligatorio</i>
Concepto Rel. :	Región

Nombre:	Persona	Fecha:	14/11/2009
Descripción:	<i>Refleja las personas que pueden ser dependientes de una o varias cuentas</i>		
Responsable:	<i>Nathalie Deppen, Paola Ricca, Diego Trías</i>		
Fuente:			
Usos:	<i>Representa las personas relacionadas con las cuentas</i>		
Supuestos:			
Identificador:	<i>[Número documento, Tipo Documento]</i>		
Atributos:			
Nombre:	<i>Nombre</i>		
Descripción:	<i>Nombre de la persona</i>		
Obligatoriedad:	<i>Este dato es obligatorio</i>		
Concepto Rel. :			
Nombre:	<i>Apellido</i>		
Descripción:	<i>Apellido de la persona</i>		
Obligatoriedad:	<i>Este dato es obligatorio</i>		
Concepto Rel. :			
Nombre:	<i>Fecha Nacimiento</i>		
Descripción:	<i>Fecha de nacimiento de la persona</i>		
Obligatoriedad:	<i>Este dato es obligatorio</i>		
Concepto Rel. :			
Nombre:	<i>Número documento</i>		
Descripción:	<i>Número del documento de identidad</i>		
Obligatoriedad:	<i>Este dato es obligatorio</i>		
Concepto Rel. :			
Nombre:	<i>Tipo documento</i>		
Descripción:	<i>Tipo de documento de identidad</i>		
Obligatoriedad:	<i>Este dato es obligatorio</i>		
Concepto Rel. :			

Nombre:	Plan	Fecha:	14/11/2009
Descripción:	<i>Refleja la categoría a la que pertenece la cuenta. Dicha categoría define las restricciones, compromisos y beneficios de la cuenta establecidos entre el vendedor de la misma (Productor) y su o sus compradores (Dependientes con Rol Titular).</i>		
Responsable:	<i>Nathalie Deppen, Paola Ricca, Diego Trías</i>		
Fuente:			

Usos:	<i>Refleja la categoría de la cuenta</i>
Supuestos:	
Identificador:	<i>Nombre</i>
Atributos:	
Nombre:	<i>Nombre</i>
Descripción:	<i>Nombre del plan</i>
Obligatoriedad:	<i>Este dato es obligatorio</i>
Concepto Rel. :	

Nombre:	Portafolio	Fecha:	14/11/2009
Descripción:	<i>Refleja los valores totales que interesan de una cuenta con respecto a sus inversiones. Dichos valores corresponden a una determinada fecha y se basan en las inversiones de la cuenta a esa fecha.</i>		
Responsable:	<i>Nathalie Deppen, Paola Ricca, Diego Trías</i>		
Fuente:			
Usos:	<i>Representa los valores totales de las inversiones de una cuenta a una determinada fecha</i>		
Supuestos:	<i>Cada valor total del portafolio está expresado en la moneda de la cuenta</i>		
Identificador:	<i>[Fecha Evaluación, Cuenta]</i>		
Atributos:			
Nombre:	<i>Fecha Evaluación</i>		
Descripción:	<i>Fecha a la cual se está evaluando el portafolio</i>		
Obligatoriedad:	<i>Este dato es obligatorio</i>		
Concepto Rel. :			
Nombre:	<i>Total Aportado</i>		
Descripción:	<i>Valor total que se ha invertido en la cuenta</i>		
Obligatoriedad:	<i>Este dato es obligatorio</i>		
Concepto Rel. :			
Nombre:	<i>Valor de rescate</i>		
Descripción:	<i>Valor total que se podría retirar en caso de cerrar la cuenta</i>		
Obligatoriedad:	<i>Este dato es obligatorio</i>		
Concepto Rel. :			
Nombre:	<i>Valor de cuenta</i>		
Descripción:	<i>Suma de los importes de todas las inversiones del portafolio y del valor en Cash Account de la cuenta</i>		
Obligatoriedad:	<i>Este dato es obligatorio</i>		
Dato calculado:	<i>Es la suma del importe de cada inversión del portafolio más el valor Cash Account del portafolio</i>		
Concepto Rel. :			
Nombre:	<i>Valor Cash Account</i>		
Descripción:	<i>Representa el valor invertido en cuentas corrientes o de plazo fijo</i>		
Obligatoriedad:	<i>Este dato es obligatorio</i>		
Concepto Rel. :			
Nombre:	<i>Tipo de Cambio</i>		
Descripción:	<i>Representa el tipo de cambio para pasar de la moneda de la cuenta</i>		

	<i>a dólares</i>
Obligatoriedad:	<i>Este dato es obligatorio</i>
Concepto Rel. :	
Nombre:	<i>Fecha Portafolio</i>
Descripción:	<i>Fecha a la cual fueron evaluados los totales. Corresponden a la mínima fecha a la cual fueron considerados los precios de las inversiones correspondientes al portafolio.</i>
Obligatoriedad:	<i>Este dato es obligatorio</i>
Concepto Rel. :	
Nombre:	<i>Inversión</i>
Descripción:	<i>Inversiones correspondientes a la evaluación de portafolio</i>
Cardinalidad:	<i>N</i>
Obligatoriedad:	<i>Este dato es obligatorio</i>
Concepto Rel. :	<i>Inversión</i>

Nombre:	Productor	Fecha:	14/11/2009
Descripción:	<i>Representa quien realiza la venta de una cuenta a una persona, quien será un dependiente con rol Titular de la misma</i>		
Responsable:	<i>Nathalie Deppen, Paola Ricca, Diego Trías</i>		
Fuente:			
Usos:	<i>Representa el vendedor de la cuenta</i>		
Supuestos:			
Identificador:	<i>Número de contrato</i>		
Atributos:			
Nombre:	<i>Nombre</i>		
Descripción:	<i>Nombre del productor</i>		
Obligatoriedad:	<i>Este dato es obligatorio</i>		
Concepto Rel. :			
Nombre:	<i>País</i>		
Descripción:	<i>País del productor</i>		
Obligatoriedad:	<i>Este dato es obligatorio</i>		
Concepto Rel. :	<i>País</i>		
Nombre:	<i>Número de contrato</i>		
Descripción:	<i>Es el número de contrato del productor con la empresa AIVA. Éste número identifica al productor.</i>		
Obligatoriedad:	<i>Este dato es obligatorio</i>		
Concepto Rel. :			

Nombre:	Región	Fecha:	14/11/2009
Descripción:	<i>Representa una región</i>		
Responsable:	<i>Nathalie Deppen, Paola Ricca, Diego Trías</i>		
Fuente:			
Usos:	<i>Refleja las regiones a la que pertenecen los países</i>		

Supuestos:	
Identificador:	<i>Nombre</i>
Atributos:	
Nombre:	<i>Nombre</i>
Descripción:	<i>Nombre de la región</i>
Obligatoriedad:	<i>Este dato es obligatorio</i>
Concepto Rel. :	

Nombre:	Rol	Fecha:	14/11/2009
Descripción:	<i>Representa el rol a través del cual una persona está relacionada con una cuenta.</i>		
Responsable:	<i>Nathalie Deppen, Paola Ricca, Diego Trías</i>		
Fuente:			
Usos:	<i>Refleja el rol de una persona en una cuenta</i>		
Supuestos:			
Identificador:	<i>Nombre</i>		
Atributos:			
Nombre:	<i>Nombre</i>		
Descripción:	<i>Nombre del rol</i>		
Obligatoriedad:	<i>Este dato es obligatorio</i>		
Concepto Rel. :			

Algoritmo Onto2rdb

A continuación se presenta un algoritmo determinístico que genera un esquema relacional de base de datos, a partir de una ontología expresada en OWL 2.0 [OWL2]. El mismo se divide en seis pasos secuenciales, los cuales tratan los diferentes escenarios que pueden darse en la ontología de entrada.

Precondiciones del algoritmo

Se asume que las clases de la ontología son disjuntas dos a dos con la excepción de subclase y superclase en las jerarquías.

Restricciones

El algoritmo que se presenta tiene en cuenta algunas de las restricciones que son especificables en la ontología:

- Totalidad en las data property (especificadas como exactly 1 en la ontología)
- Identificadores de las clases(especificados con el axioma HasKey de OWL2 [OWL2])

Al terminar el proceso se genera un log con las restricciones que forman parte de la ontología y no han sido tenidas en cuenta en la generación del esquema relacional, estas restricciones pueden ser programadas como triggers o asserts en la base de datos o en los programas que cargan la misma (dependiendo de qué posibilidades brinde el motor de base de datos que se esté utilizando y las preferencias del grupo de desarrollo)

Algoritmo

Nuevo esquema

Se crea un esquema de base de datos vacío para agregarle las tablas en los siguientes pasos.

Una tabla por cada clase

Se recorren las clases alfabéticamente por su nombre y por cada una, se agrega al nuevo esquema de base de datos (creado en el paso anterior), una tabla con el mismo nombre de la clase. Para la nueva tabla se agrega un campo por cada datatype property de la clase con el mismo nombre y tipo (rango de la property) de esta. Si existe una restricción en la ontología de exactly 1 para esa datatype property, el campo se agrega no admitiendo nulos, de otra forma, el campo admitirá nulos.

Claves, relaciones 1 a 1 y relaciones n a 1

Mientras haya alguna tabla que no tiene clave primaria definida o queden relaciones N:1 o 1:1 (mas adelante se explica cuales son las condiciones para que una relación sea N:1 o 1:1) que no se han reflejado en la base de datos, se ejecutan iterativamente las siguientes 3 acciones:

1.1. Claves

Se recorren alfabéticamente las clases que cumplen:

- En la tabla de mismo nombre, no hay clave primaria definida

- Todas las propiedades utilizadas en la cláusula HasKey son data property o se utilizan object properties tales que ya se agregó a la tabla antes mencionada el campo Identificador de la tabla que representa la clase recorrido de la relación (identificador que representa la relación).

Para cada clase que cumple con las condiciones antes mencionada se tienen dos posibilidades:

- 1.1.1. Si la clase tiene un solo atributo definido como key (utilizando el axioma HasKey de OWL 2.0 [OWL2]) se indica como clave primaria en la tabla (clave natural).
- 1.1.2. Si la clase no tiene atributos definidos como key o tiene más de uno, se agrega un campo a la tabla llamado como la clase agregándole el prefijo "Id" de tipo entero auto incremental y se indica como clave primaria de la tabla (clave sustituta). Si hay más de un atributo definido como key, se agrega la restricción de unicidad (índice único) sobre los campos sobre los que está definida la key.

1.2. Relaciones n a 1

Para todas las relaciones de la ontología que cumplen:

- Es una relación funcional
 - Su relación inversa no está definida o es no funcional
- (Con estas dos condiciones se tiene una relación de N:1 entre el dominio y rango de la relación)
- La tabla de mismo nombre que el recorrido de la relación ya tiene su clave definida.

Puede agregarse la clave foránea que representa la relación. Para esto:

- 1.2.1. Ordenar alfabéticamente las relaciones que cumplen la condición antes mencionada.
- 1.2.2. Dadas las clases dominio y rango de la relación (A y B de ahora en más):
 - i. Agregar en la tabla A de la base de datos, un campo con el mismo nombre de la relación y el mismo tipo que el campo clave de la tabla B. Este campo no puede admitir nulos.
 - ii. Agregar la restricción de integridad referencial:
Para cada valor de $x = A.Relacion$ debe existir un registro de B tal que $x = B.ClaveB$.

NOTA: Se denomina "Relación" al campo agregado en la tabla A.

En Figura 88 y Figura 87 se muestra el ejemplo de la parte de la ontología y su representación relacional respectivamente.

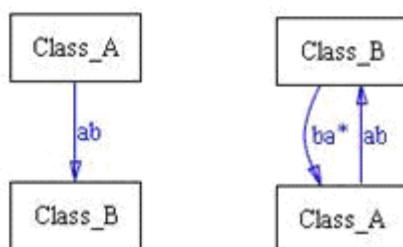


Figura 88. Relación N:1 en la ontología

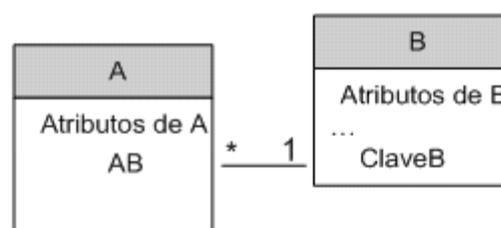


Figura 87. Representación relacional de la Relación N:1 en la ontología

1.3. Relaciones 1 a 1

Para todas las relaciones de la ontología que cumplen:

- Es una relación funcional
- Su relación inversa es funcional

(Con estas dos condiciones se tiene una relación de 1:1 entre el dominio y rango de la relación)

- Tiene definida la clave de la tabla que representa al recorrido

Puede agregarse la clave foránea que representa la relación. Para esto:

1.3.1. Ordenar alfabéticamente las relaciones que cumplen la condición antes mencionada.

1.3.2. Para cada relación, dadas las clases dominio y rango de la relación (A y B de ahora en más),

- Si su inversa ya fue procesada como una relación 1:1 con este algoritmo, volver al paso 3.3.2 para continuar con la siguiente.
- Agregar en la tabla que se lista primera de las dos en orden alfabético (de ahora en más A), un campo con el nombre de la relación.
- Agregar la restricción de integridad referencial:
Para cada valor de $x = A.Relacion$ debe existir un registro de B tal que $x = B.ClaveB$
- Agregar una restricción unicidad (índice único) sobre el campo A.Relacion.

NOTA: Se denomina “ClaveB” al campo que es clave de la tabla B y “Relación” al campo agregado en la tabla A.

En Figura 90 y Figura 89 se muestra el ejemplo de la parte de la ontología y su representación relacional respectivamente.

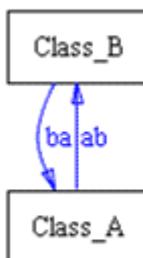


Figura 90. Relación 1:1 en la ontología

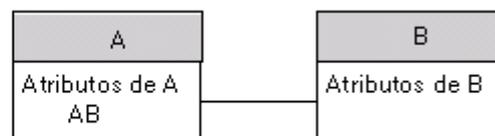


Figura 89. Representación relacional de la relación 1:1 de la ontología

Relaciones n a n

Para todas las relaciones de la ontología que cumplen:

- No es una relación funcional
- Su relación inversa es no funcional o no existe

(Con estas dos condiciones se tiene una relación de N:N entre el dominio y rango de la relación)

- 1.4. Ordenar alfabéticamente las relaciones que cumplen las condiciones antes mencionada.
- 1.5. Dadas las clases dominio y rango de la relación (A y B de ahora en más)
 - i. Si su inversa ya fue procesada como una relación N:N con este algoritmo, volver al paso 4.2 para continuar con la siguiente.
 - ii. Agregar a la base de datos una tabla con el nombre de la relación
 - iii. Agregar a dicha tabla el campo que corresponde a la clave de la tabla A y el campo que corresponde a la clave de la tabla B
 - iv. Agregar las restricciones de integridad referencial a las dos tablas.
 - v. Identificar la clave de la nueva tabla AB, la cual estará compuesta por los dos campos agregados anteriormente.

En Figura 92 y Figura 91 se muestra el ejemplo de la parte de la ontología y su representación relacional respectivamente.

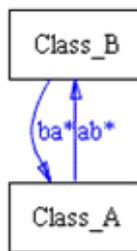


Figura 92. Relación N:N en la ontología

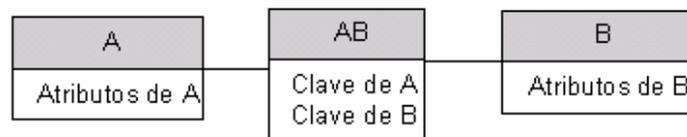


Figura 91. Representación relacional de la Relación N:N en la ontología

Jerarquías

Para cada jerarquía de la ontología (isa), se tendrá una especialización en la base de datos. Por tanto, para cada jerarquía se realizará lo siguiente:

- 1.6. Agregar a la tabla nombrada igual que la subclase (B de ahora en más) el campo correspondiente a la clave de la tabla nombrada igual que la superclase (A de ahora en más), este campo no admite nulos.
- 1.7. Agregar la respectiva restricción de integridad referencial (B.ClaveA incluido en A.ClaveA).

En Figura 94 y Figura 93 se muestra el ejemplo de la parte de la ontología y su representación relacional respectivamente.



Figura 94. Jerarquía en la ontología

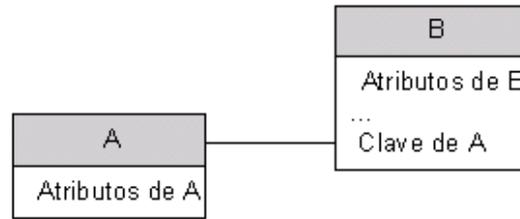


Figura 93. Representación relacional de la jerarquía en la ontología

Log de restricciones no consideradas

Se recorren las restricciones de todas las clases, y para cada restricción que no ha sido considerada (que no sea exactly 1 de una datatype property y no sea hasKey) se lista la clase y a continuación la restricción.

Con esto se deja al equipo de desarrollo la posibilidad de ver rápidamente que restricciones han sido dejadas de lado para que puedan tomar las medidas correspondientes, que (como ya fue explicado antes), puede ser programarlas en la base de datos o en los programas que cargan la misma.

Guía Resumida

Este trabajo propone guías que apoyan el proceso de construcción de un DW desde relevar los requerimientos, representar la realidad de la empresa, generar el diseño conceptual, el diseño lógico, hasta la carga de datos necesaria para el correcto funcionamiento del sistema.

La guía propuesta consta de cinco fases presentadas en la Figura 95:

1. Entender la realidad y documentarla en Metadata de conceptos.
2. Modelar formalmente la realidad mediante una ontología.
3. Generar el modelo conceptual expresado en CMDM.
4. Crear el ODS.
5. Generar el diseño lógico y la carga de datos.

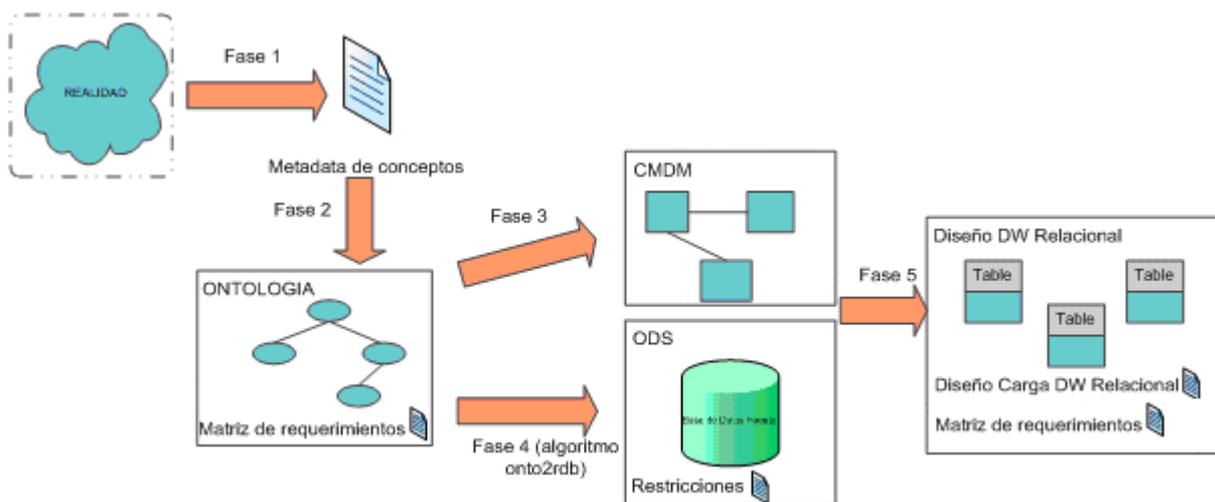


Figura 95. Fases de la guía de construcción de un DW

Se detalla a continuación cada una de las fases propuestas, indicando para cada una:

- **Objetivo:** descripción del objetivo principal de la fase
- **Entrada:** elementos requeridos para inicializar la fase
- **Salida:** elementos resultados de la aplicación de lo especificado en la fase
- **Participantes:** se consideran tres conjuntos no disjuntos de participantes:
 - **Expertos de la realidad:** en general el cliente, quien conoce el negocio y la realidad sobre la cual se está trabajando
 - **Analistas:** integrantes del equipo de desarrollo, quienes se encargan de realizar el análisis de los requerimientos y la realidad sobre la cual se definen.
 - **Diseñadores:** integrantes del equipo de desarrollo, quienes se encargan de realizar el diseño de la solución a los requerimientos.
- **Descripción:** especificación del trabajo que debe realizarse en la fase descrita
- **Postcondición:** contexto en el cual se encuentra al finalizar la fase. Si la Postcondición no se cumple, en cualquiera de las fases en que se esté se debe analizar el origen del error, y volver atrás para subsanarlo, repasando las siguientes fases para revisar que los siguientes resultados sean correctos o corregirlos si no los son.

Fase 1 - Entender y documentar la realidad

Objetivo

El objetivo de esta fase es comprender y documentar los conceptos de la realidad y la relación entre los mismos, documentando de esta manera el conocimiento tácito que la organización cliente posee en conocimiento explícito documentado. Un segundo objetivo de esta fase es analizar y documentar los requerimientos.

Entrada

Requerimientos iniciales

Salida

Como salida de esta fase se tienen dos documentos:

- Metadata de conceptos (ver Anexo I)
- Documento con los Requerimientos del sistema

Participantes

- Expertos de la realidad
- Analistas del equipo de desarrollo.

Descripción

Esta fase es guiada por los requerimientos iniciales, para llevarla a cabo se deben mantener reuniones con los expertos en la realidad y con los expertos en el sistema transaccional, también suelen ser de utilidad manuales de usuario de dicho sistema o cualquier documentación del mismo.

Los conceptos que se identifiquen durante esta fase deben ser definidos de forma clara y concisa, y documentados en Metadata de conceptos, así como las relaciones entre los mismos. Los requerimientos relevados durante esta fase deben ser registrados en la terminología que se ha documentado (en Metadata de conceptos).

Uno de los objetivos básicos de esta fase es que todos los involucrados se pongan de acuerdo en los términos y significados de los mismos, para así no tener mal entendidos ni la necesidad de aclarar el significado de los términos.

Postcondición

Al terminar esta fase se debe tener:

- Documento con todos los requerimientos del sistema
- Documento Metadata de conceptos que contenga todos y cada uno de los conceptos que se utilizan en algún requerimiento.

Fase 2 - Modelado y formalización del dominio

Objetivo

El objetivo de esta fase es formalizar lo relevado en la fase I, para así poder revisar la consistencia del modelo antes de seguir adelante. Lo que debe hacerse en esta fase es modelar lo documentado en el artefacto Metadata de conceptos.

Entrada

- Metadata de conceptos
- Documento con requerimientos relevados en la fase I

Salida

Como salida de esta fase se tiene un modelo semántico formal consistente. Se recomienda para esto una ontología owl2 [OWL2].

Participantes

- Analistas del equipo de desarrollo.
- Expertos en la realidad, pero solo para dudas puntuales.

Descripción

En base al documento Metadata de conceptos, se construye un modelo de dominio formal. Nótese que se habla de un modelo de dominio, por lo que los conceptos en este modelo deben ser parte de la realidad y no objetos informáticos, de esta forma el modelo puede ser entendido y criticado por expertos de la realidad. Es necesario indicar con la cláusula HasKey de [OWL2] los atributos que identifican a un concepto.

Es importante destacar que en el caso de hacerse cambios en alguno de los conceptos como resultado del análisis o de la crítica de un experto en la realidad, los mismos deben plasmarse en el documento Metadata de conceptos, para mantener la documentación actualizada a lo largo de todo el proceso.

Además de crear el modelo de dominio, en esta fase se genera la matriz de requerimientos. Este documento brinda trazabilidad entre los requerimientos y los conceptos, por lo que ante un cambio en los requerimientos (o en los conceptos) se puede rápidamente conocer su impacto. En la Figura 96 se muestra el documento que soporta la matriz de requerimientos.

Concepto	R1	R2	...	RN
<i>Clases</i>				
Concepto1	✓			
Concepto2	✓			
...	✓			
ConceptoN	✓			
<i>Relaciones</i>				
Relación1	✓			
...	✓			
RelaciónX	✓			
<i>Herencia</i>				
ConceptoL "is a" ConceptoJ	✓			
.....	✓			

Figura 96. Matriz de requerimientos

Para construir la matriz simplemente se disponen todos los conceptos en el eje "Y" y todos los requerimientos en el eje "X" y se marca en la intersección cuando un concepto es utilizado en un requerimiento.

Postcondición

Para finalizar esta fase se debe cumplir:

1. Todos los conceptos que estén involucrados en un requerimiento deben formar parte de la ontología
2. La ontología es consistente (no hay contradicciones)

3. El documento Metadata de conceptos es coherente con la ontología (si se han hecho cambios a los conceptos, los mismos se impactaron)
4. La matriz de requerimientos contempla todos los requerimientos y todos los conceptos

Fase 3 - Construcción del modelo conceptual

Objetivo

El objetivo de la presente fase es la construcción del modelo conceptual multidimensional del DW a partir del modelo de dominio. Para cumplir el objetivo se siguen las once reglas que se presentan en la descripción de la presente fase.

Entrada

- Metadata de conceptos
- Matriz de requerimientos
- Modelo de dominio (ontología)
- Conjunto de requerimientos iniciales

Salida

Al finalizar esta fase, se contará con el modelo conceptual multidimensional del DW, expresado en CMDM.

Participantes

- Analistas del equipo de desarrollo
- Diseñadores del equipo de desarrollo

Descripción

A continuación se presentan las once reglas que se deben seguir para construir el modelo CMDM que es la salida de esta fase.

Definición de niveles

Regla I

REGLA R1 – Clases a mapearse como niveles en CMDM
<p>Objetivo: Identificar las clases a partir de las cuales se definirán los niveles del modelo conceptual multidimensional</p>
<p>Entrada:</p> <ul style="list-style-type: none">• Ontología de dominio• Matriz de requerimientos
<p>Descripción: El conjunto inicial de clases candidatas C está formada por las clases definidas en la matriz de requerimientos.</p> <p>Para cada clase del conjunto C se identifican sus restricciones propias (no heredadas). Se agrega al conjunto C las clases correspondientes al rango de cada una de las propiedades participantes en las restricciones antes identificadas.</p> <p>Para cada una de las clases del conjunto C se identifica si la misma pertenece a una categorización en la ontología. En caso que se cumpla esta condición todas sus superclases se agregan al conjunto C.</p> <p>Las clases del conjunto C que en la ontología tengan al menos una propiedad que no sea heredada, son consideradas a mapearse como niveles en CMDM.</p>
<p>Resultado: Conjunto de clases a mapearse como niveles del modelo multidimensional CMDM.</p>

Regla II

REGLA R2 – Definición de niveles en CMDM
<p>Objetivo: Definir el conjunto de niveles del modelo CMDM.</p>
<p>Entrada:</p> <ul style="list-style-type: none">• Conjunto de clases a mapearse como niveles.• Ontología de dominio.
<p>Descripción: Para cada una de las clases del conjunto de clases a mapearse como niveles del modelo multidimensional, se identifica en la ontología su nombre y las propiedades definidas como datos simples (Data Properties). Cada uno de los niveles se nombrará según el nombre que la clase tenga definido en la ontología. Los atributos de los niveles quedarán determinados por las propiedades definidas como propias, considerando solo los datos simples.</p>
<p>Resultado: Conjunto de niveles del modelo CMDM.</p>

Definición de jerarquías

Regla III

REGLA R3 – Definición de cardinalidades
<p>Objetivo: Identificar y definir cardinalidades en las relaciones interclases de la ontología.</p>
<p>Entrada:</p> <ul style="list-style-type: none">• Conjunto de clases a mapearse como niveles.• Ontología de dominio.
<p>Descripción: Identificar las relaciones existentes entre las clases del conjunto de clases a mapearse como niveles.</p> <p>En las relaciones para las cuales no se haya definido una restricción de cardinalidad se considera que la cardinalidad asociada desde la clase dominio a la clase rango es N.</p> <p>Si una relación no tiene definida una relación inversa, la cardinalidad definida para esta relación inversa se asume como cardinalidad N, en dirección rango-dominio.</p> <p>La cardinalidad entre dos clases queda definida por la cardinalidad de la relación entre estas dos clases y la cardinalidad de su relación inversa.</p>
<p>Resultado: Conjunto de cardinalidades entre las clases candidatas a mapearse como niveles.</p>

Regla IV

REGLA R4 – Identificación de orden entre niveles a partir de cardinalidad N:1
<p>Objetivo: Identificar ordenes entre los niveles del modelo multidimensional.</p>
<p>Entrada:</p> <ul style="list-style-type: none">• Ontología de dominio.• Conjunto de relaciones identificadas con cardinalidad N:1.• Conjunto de niveles.
<p>Descripción: Para cada relación del conjunto de relaciones identificadas con cardinalidad N:1, se define un orden entre dos niveles donde la clase que participa con cardinalidad N se mapea como nivel inferior y la clase que participa con cardinalidad 1 se mapea como nivel superior. De esta manera se obtiene un orden entre los niveles en CMDM.</p>
<p>Resultado: Conjunto de niveles parcialmente ordenados.</p>

Regla V

REGLA R5 – Identificación de orden entre niveles a partir de cardinalidad 1:N

Objetivo:

Identificar ordenes entre los niveles del modelo multidimensional.

Entrada:

- Ontología de dominio.
- Conjunto de relaciones identificadas con cardinalidad 1:N.
- Conjunto de niveles.

Descripción:

Para cada relación del conjunto de relaciones identificadas con cardinalidad 1:N, se define un orden entre dos niveles donde la clase con cardinalidad 1 se mapea como nivel superior y la clase que participa con cardinalidad N se mapea como nivel inferior. De esta manera se obtiene un orden entre los niveles en CMDM.

Resultado:

Conjunto de niveles parcialmente ordenados.

Regla VI

REGLA R6 – Identificación de orden entre niveles a partir de cardinalidad N:N

Objetivo:

Identificar niveles que no pertenecen a una misma jerarquía del modelo multidimensional.

Entrada:

- Ontología de dominio.
- Conjunto de relaciones identificadas con cardinalidad N:N.
- Conjunto de niveles.

Descripción:

Esta regla define cuando dos niveles no pueden ser parte de una misma jerarquía. Las clases que definen estas relaciones son independientes entre sí, no existiendo características por la cual sea posible agruparlas.

Es posible considerar dos posibles casos:

- las clases se mapean como niveles de jerarquías diferentes en una misma dimensión.
- las clases se mapean como niveles de diferentes dimensiones.

Esta decisión depende del caso en particular que se esté resolviendo pero las clases nunca podrán mapearse como niveles de una misma jerarquía.

Resultado:

Conjunto de niveles pertenecientes a diferentes jerarquías de una dimensión o niveles de dimensiones diferentes en el modelo multidimensional.

Regla VII

REGLA R7 – Identificación de orden entre niveles a partir de relaciones 1:1 con Totalidad

Objetivo:

Identificar ordenes entre los niveles del modelo multidimensional.

Entrada:

- Ontología de dominio.
- Conjunto de niveles identificados que participan en una relación con cardinalidad 1:1 con totalidad.

Descripción:

Dada la relación que cumple con cardinalidad 1:1 y cuenta con la propiedad de totalidad, ya sea para la clase dominio o para la clase rango, se construye la jerarquía mapeando la clase para la cual se cumple la totalidad como nivel superior y la clase restante conforma el nivel inferior de la misma.

De este modo, al realizar las operaciones entre niveles, es posible obtener elementos que cumplan las características dadas, descartando aquellos que no lo hagan.

Resultado:

Conjunto de niveles parcialmente ordenados.

Regla VIII

REGLA R8 – Identificación de orden entre niveles a partir de cardinalidad 1:1

Objetivo:

Identificar ordenes entre niveles del modelo multidimensional.

Entrada:

- Ontología de dominio.
- Conjunto de relaciones identificados con cardinalidad 1:1 (sin totalidad).
- Conjunto de niveles.

Descripción:

Cualquiera de las clases participantes en la cardinalidad puede ser considerada como nivel superior o inferior, quedando a criterio del diseñador definir el orden entre los niveles. Esta decisión depende del caso en particular que se esté resolviendo.

Resultado:

Conjunto de niveles parcialmente ordenados.

Regla IX

REGLA R9 – Categorización de subclases (a partir de subclases disjuntas)
<p>Objetivo: Identificar ordenes entre niveles del modelo multidimensional.</p>
<p>Entrada:</p> <ul style="list-style-type: none">• Ontología de dominio.• Conjunto de clases a mapearse como niveles.
<p>Descripción: Para cada clase del conjunto de clases a mapearse como niveles, identificar aquellas que son superclases y cuyas subclases sean disjuntas.</p> <p>Dada una superclase del conjunto, pueden obtenerse un nuevo orden entre niveles donde la superclase se mapea como nivel inferior y como nivel superior se crea un nivel que agrupe todas las subclases. Al nivel superior se le asigna un nombre que identifique la agrupación, por ejemplo TipoCategoría. Para este nivel se define un atributo que permita identificar cada uno de los elementos que lo conforman.</p>
<p>Resultado: Conjunto de niveles parcialmente ordenados.</p>

Definición de dimensiones y relaciones dimensionales

Regla X

REGLA R10 – Identificación de dimensiones en CMDM
<p>Objetivo: Identificar dimensiones del modelo multidimensional.</p>
<p>Entrada:</p> <ul style="list-style-type: none">• Conjunto de niveles.• Conjunto de niveles parcialmente ordenados.
<p>Descripción: Identificar las tuplas de niveles ordenados las cuales tengan algún nivel en común. Una dimensión queda formada por el subconjunto de tuplas de niveles previamente ordenados que tengan algún nivel en común, manteniendo el orden parcial definido entre ellos. Las dimensiones deben tener un nivel mínimo donde exista un identificador para cada elemento de la misma. Gráficamente la dimensión se forma uniendo las tuplas ordenadas según el nivel en común, manteniendo el orden definido. Todos los niveles deben formar parte de alguna dimensión. Puede ocurrir que exista una dimensión con un único nivel.</p> <p>Si como resultado de lo anterior se obtienen dimensiones con más de una jerarquía, es tarea del diseñador, y en base a los requerimientos, decidir si lo que necesita es una dimensión con más de una jerarquía o varias dimensiones de una jerarquía.</p>
<p>Resultado: Conjunto de dimensiones del modelo multidimensional CMDM.</p>

Regla XI

REGLA R11 – Identificación de relaciones dimensionales en CMDM
<p>Objetivo: Identificar relaciones dimensionales del modelo multidimensional CMDM.</p>
<p>Entrada:</p> <ul style="list-style-type: none">• Conjunto de dimensiones donde su nivel mínimo forme parte de más de una dimensión o a partir de su nivel mínimo se desprendan varias jerarquías.
<p>Descripción:</p> <p>En el caso de contar con una dimensión donde a partir de su nivel inferior <i>I</i> se desprendan varias jerarquías, y a su vez la cardinalidad entre <i>I</i> con los niveles superiores sea N:1 en todos los casos, es posible definir una relación dimensional donde <i>I</i> sea el eje central de la relación. Las dimensiones que definen dicha relación dimensional quedan formadas por el conjunto de niveles superiores a partir del eje central.</p> <p>En el caso de contar con un conjunto de dimensiones donde su nivel inferior <i>I</i> es común a todas, y a su vez la cardinalidad entre <i>I</i> y los niveles superiores sea N:1 en todos los casos, es posible definir una relación dimensional donde <i>I</i> sea el eje central de la relación. Las dimensiones que definen dicha relación dimensional quedan formadas por el conjunto de niveles superiores a partir del eje central.</p> <p>Esta consideración queda en manos del diseñador, ya que debe ser él quien defina el eje central de la discusión.</p>
<p>Resultado: Relaciones dimensionales del modelo multidimensional.</p>

Postcondición

Al terminar esta fase se debe tener un modelo conceptual CMDM el cual satisface todos los requerimientos relevados en la fase 1.

Fase 4 - Construcción del ODS

Objetivo

Esta fase tiene como objetivo generar una fuente de datos relacional única que cubre toda la realidad.

Entrada

- Ontología OWL2 [OWL2] creada en la fase II.

Salida

- Base de datos relacional que puede almacenar toda la información del DW

Participantes

- Analistas del equipo de desarrollo.
- Diseñadores del equipo de desarrollo

Descripción

Esta fase es cumplida mediante la aplicación del algoritmo. Este algoritmo genera la estructura de una base de datos relacional a partir de la ontología que tiene como entrada. En la base de datos generada se almacenaran los datos recabados de todas las fuentes de datos, mediante la

carga con las ETL (no considerado en este proyecto). Ver detalle del algoritmo en el anexo Algoritmo Onto2rdb.

Precondiciones del algoritmo

Se asume que las clases de la ontología son disjuntas dos a dos con la excepción de subclase y superclase en las jerarquías.

Restricciones

El algoritmo que se presenta tiene en cuenta algunas de las restricciones que son especificables en la ontología:

- Totalidad en las data property (especificadas como exactly 1 en la ontología)
- Identificadores de las clases(especificados con el axioma HasKey de OWL2 [OWL2])

Al terminar el proceso se genera un log con las restricciones que forman parte de la ontología y no han sido tenidas en cuenta en la generación del esquema relacional, estas restricciones pueden ser programadas como triggers o asserts en la base de datos o en los programas que cargan la misma (dependiendo de qué posibilidades brinde el motor de base de datos que se esté utilizando y las preferencias del grupo de desarrollo)

Algoritmo

Nuevo esquema

Se crea un esquema de base de datos vacío para agregarle las tablas en los siguientes pasos.

Una tabla por cada clase

Se recorren las clases alfabéticamente por su nombre y por cada una, se agrega al nuevo esquema de base de datos (creado en el paso anterior), una tabla con el mismo nombre de la clase. Para la nueva tabla se agrega un campo por cada datatype property de la clase con el mismo nombre y tipo (rango de la property) de esta. Si existe una restricción en la ontología de exactly 1 para esa datatype property, el campo se agrega no admitiendo nulos, de otra forma, el campo admitirá nulos.

Claves, relaciones 1 a 1 y relaciones n a 1

Mientras haya alguna tabla que no tiene clave primaria definida o queden relaciones N:1 o 1:1 (mas adelante se explica cuales son las condiciones para que una relación sea N:1 o 1:1) que no se han reflejado en la base de datos, se ejecutan iterativamente las siguientes 3 acciones:

1.1. Claves

Se recorren alfabéticamente las clases que cumplen:

- En la tabla de mismo nombre, no hay clave primaria definida
- Todas las propiedades utilizadas en la cláusula Haskey son data property o se utilizan object properties tales que ya se agregó a la tabla antes mencionada el campo Identificador de

la tabla que representa la clase recorrido de la relación (identificador que representa la relación).

Para cada clase que cumple con las condiciones antes mencionada se tienen dos posibilidades:

- 1.1.1. Si la clase tiene un solo atributo definido como key (utilizando el axioma HasKey de OWL 2.0 [OWL2]) se indica como clave primaria en la tabla (clave natural).
- 1.1.2. Si la clase no tiene atributos definidos como key o tiene más de uno, se agrega un campo a la tabla llamado como la clase agregándole el prefijo "Id" de tipo entero auto incremental y se indica como clave primaria de la tabla (clave sustituta). Si hay más de un atributo definido como key, se agrega la restricción de unicidad (índice único) sobre los campos sobre los que está definida la key.

1.2. Relaciones n a 1

Para todas las relaciones de la ontología que cumplen:

- Es una relación funcional
 - Su relación inversa no está definida o es no funcional
- (Con estas dos condiciones se tiene una relación de N:1 entre el dominio y rango de la relación)
- La tabla de mismo nombre que el recorrido de la relación ya tiene su clave definida.

Puede agregarse la clave foránea que representa la relación. Para esto:

- 1.2.1. Ordenar alfabéticamente las relaciones que cumplen la condición antes mencionada.
- 1.2.2. Dadas las clases dominio y rango de la relación (A y B de ahora en más):
 - iii. Agregar en la tabla A de la base de datos, un campo con el mismo nombre de la relación y el mismo tipo que el campo clave de la tabla B. Este campo no puede admitir nulos.
 - iv. Agregar la restricción de integridad referencial:
Para cada valor de $x = A.Relacion$ debe existir un registro de B tal que $x = B.ClaveB$.

NOTA: Se denomina "Relación" al campo agregado en la tabla A.

1.3. Relaciones 1 a 1

Para todas las relaciones de la ontología que cumplen:

- Es una relación funcional
 - Su relación inversa es funcional
- (Con estas dos condiciones se tiene una relación de 1:1 entre el dominio y rango de la relación)
- Tiene definida la clave de la tabla que representa al recorrido

Puede agregarse la clave foránea que representa la relación. Para esto:

- 1.3.1. Ordenar alfabéticamente las relaciones que cumplen la condición antes mencionada.
- 1.3.2. Para cada relación, dadas las clases dominio y rango de la relación (A y B de ahora en más),

- v. Si su inversa ya fue procesada como una relación 1:1 con este algoritmo, volver al paso 3.3.2 para continuar con la siguiente.
- vi. Agregar en la tabla que se lista primera de las dos en orden alfabético (de ahora en más A), un campo con el nombre de la relación.
- vii. Agregar la restricción de integridad referencial:
Para cada valor de $x = A.Relacion$ debe existir un registro de B tal que $x = B.ClaveB$
- viii. Agregar una restricción unicidad (índice único) sobre el campo A.Relacion.

NOTA: Se denomina “ClaveB” al campo que es clave de la tabla B y “Relación” al campo agregado en la tabla A.

Relaciones n a n

Para todas las relaciones de la ontología que cumplen:

- No es una relación funcional
- Su relación inversa es no funcional o no existe

(Con estas dos condiciones se tiene una relación de N:N entre el dominio y rango de la relación)

- 1.4. Ordenar alfabéticamente las relaciones que cumplen las condiciones antes mencionada.
- 1.5. Dadas las clases dominio y rango de la relación (A y B de ahora en más)
 - vi. Si su inversa ya fue procesada como una relación N:N con este algoritmo, volver al paso 4.2 para continuar con la siguiente.
 - vii. Agregar a la base de datos una tabla con el nombre de la relación
 - viii. Agregar a dicha tabla el campo que corresponde a la clave de la tabla A y el campo que corresponde a la clave de la tabla B
 - ix. Agregar las restricciones de integridad referencial a las dos tablas.
 - x. Identificar la clave de la nueva tabla AB, la cual estará compuesta por los dos campos agregados anteriormente.

Jerarquías

Para cada jerarquía de la ontología (isa), se tendrá una especialización en la base de datos. Por tanto, para cada jerarquía se realizar lo siguiente:

- 1.6. Agregar a la tabla nombrada igual que la subclase (B de ahora en más) el campo correspondiente a la clave de la tabla nombrada igual que la superclase (A de ahora en más), este campo no admite nulos.
- 1.7. Agregar la respectiva restricción de integridad referencial (B.ClaveA incluido en A.ClaveA).

Log de restricciones no consideradas

Se recorren las restricciones de todas las clases, y para cada restricción que no ha sido considerada (que no sea exactly 1 de una datatype property y no sea hasKey) se lista la clase y a continuación la restricción.

Con esto se deja al equipo de desarrollo la posibilidad de ver rápidamente que restricciones han sido dejadas de lado para que puedan tomar las medidas correspondientes, que (como ya fue explicado antes), puede ser programarlas en la base de datos o en los programas que cargan la misma.

Postcondición

Al terminar esta fase se debe tener una base de datos (denominada fuente canónica) cuyo esquema permite almacenar instancias de todos los conceptos representados en la ontología.

Fase 5 - Generación del diseño lógico y la carga de datos

Objetivo

Esta fase tiene como objetivo generar el diseño lógico y la carga del DW a partir de la fuente canónica y el modelo CMDM.

Entrada

- Base de datos relacional que puede almacenar toda la información del DW (obtenido en la Fase IV).
- Modelo multidimensional representado en CMDM (obtenido en la Fase III).
- Matriz de requerimientos de la Fase II

Salida

- Vista unificada para cada fragmento de dimensión o franja de cubo.
- Matriz de trazabilidad completa (Conceptos, Requerimientos, Cubos)

Participantes

- Analistas del equipo de desarrollo.
- Expertos en la realidad, pero solo para dudas puntuales.
- Diseñadores del equipo de desarrollo.

Descripción

Algoritmo para el proceso de carga:

En los primeros cuatro etapas se define todo lo necesario para aplicar el algoritmo en la etapa E.5.

E1: Esquema conceptual

Se toma el modelo conceptual en CMDM, se define una clave por cada nivel y se cambian aquellos nombres de atributos que aparecen repetidos.

El resultado de este paso son las dimensiones (con los cambios antes descriptos) y las relaciones dimensionales del modelo en CMDM.

E2: Base de datos fuente

Se describe la base de datos canónica en la cual se aplicará el algoritmo.

E3: Lineamientos

Para cada relación dimensional, definir los cubos que se van a construir y cuál será el detalle de información de cada uno. Además se deben definir las franjas de información para cada cubo (año actual, anteriores, clientes uruguayos, clientes extranjeros, etc.). Y se deben establecer las estrategias de diseño, lo que implica definir fragmentos en cada dimensión. Fragmentar la dimensión es agrupar niveles de la misma.

E4: Correspondencias.

Se encuentran las correspondencias entre la BDC y el modelo en CMDM. Si existen atributos calculados, se definen en este paso.

En este momento se cuenta con todo lo necesario para aplicar el algoritmo definido, por lo que se pasa al paso:

E5: Algoritmo

El algoritmo se divide en 4 partes con un total de 15 pasos:

- I. Pasos 1- 6 para armar las tablas de dimensiones
- II. Pasos 7-12 para armar los cubos con mapeo simple
- III. Pasos 13-14 para armar los cubos con mapeo recursivo
- IV. Paso 15 para armar las franjas de cubos

I. Tratamiento de Dimensiones

- Paso 1: Construcción de Esqueletos:

Tomar cada fragmento y armar una única vista que lo mapee, definiendo así el esqueleto de cada dimensión con los atributos de mapeo directo.

- Paso 2: Nombres Atributos

Generar vistas si existen ítems del fragmento con correspondencia directa que tengan distinto nombre en la relación de la BDF de la que provienen (nombre(ítem) <> nombre(atributo)). Se hace el cambio de nombre directamente en la vista resultado del paso 1.

- Paso 3: Correspondencias

Generar atributos para ítems con mapeo calculado o externo. Para esto agrega cada nuevo atributo en la vista resultado del paso 2. Para seguir con una única vista, propone utilizar subconsultas en el FROM para resolver los atributos resultados de operaciones de agrupamiento.

- Paso 4: Aplicar Filtros

Modificar las vistas para cubrir la necesidad de que las instancias deseadas para el DW cumplan ciertas restricciones (f(Ci1, ..., Cik)), expresadas como condiciones sobre los datos de la BDF. Los filtros se agregan directamente en la vista resultado del paso 3.

Este paso aplica únicamente en el tratamiento de cubos, no en el tratamiento de dimensiones

- Paso 5: Eliminar Atributos

Aplicar esta regla si existen atributos en el resultado parcial que no están dentro de las correspondencias definidas, es decir, que no deben estar en el DW.

Este paso aplica únicamente en el tratamiento de cubos, no en el tratamiento de dimensiones.

- Paso 6: Claves

Finalmente para cada fragmento cuya clave difiera de la clave en el DW resultado, definir la clave necesaria.

II. Tratamiento de Cubos

- Paso 7 al 12:

Estos pasos, donde se tratan los cubos definidos en función de la BDF, simplemente vuelven a ejecutar los pasos del 1 al 6 vistos anteriormente. O sea, se construye una secuencia similar de vistas, con la diferencia de que se toman las correspondencias de cubos, en lugar de las de fragmentos.

Una de las mayores diferencias se da en el paso 11, donde se eliminan los atributos sin correspondencia (similar al paso 5), se considera la existencia de medidas y por esto cambia la regla aplicada.

- Paso 11: Eliminar Atributos en Cubos

La diferencia radica en que no se puede realizar la simplificación hecha en el Paso 5.

III. Cubos Recursivos

Se analizan los cubos con mapeo recursivo, o sea aquellos cubos que se basan en otros cubos. En estos casos, se buscará reducir el detalle de una dimensión, eliminando la misma (Drill-up total) o bien subiendo por niveles de la jerarquía dentro de una dimensión (Drill-up parcial).

- Paso 13: **Fragmentos Ficticios**

En una primera (paso 13) se construyen fragmentos de dimensión denominados **Fragmentos Ficticios**, dado que sólo se utilizan para construir los cubos recursivos.

Para esto, se ejecutan nuevamente los pasos 1 al 6 del algoritmo por lo cual las vistas que se construyen son equivalentes a las ya vistas para estos pasos.

- Paso 14:

Se utilizan estos fragmentos para efectuar las operaciones de Drill-Up sobre el cubo base y así obtener el cubo recursivo. Estos fragmentos ficticios se construyen sólo para las dimensiones en que sea necesario hacer un **Drill-Up Parcial**, donde esto significa no eliminar la dimensión en el cubo sino ir a un nivel superior.

Dichos fragmentos también están especialmente diseñados para contener sólo los niveles origen y el de destino del Drill-Up.

En este paso se aplican las operaciones de Drill-Up al cubo base para obtener el cubo recursivo, con la particularidad de que las operaciones de Drill-Up se hacen en una dimensión a la vez.

IV. Franjas

- Paso 15:

Se aplicará en caso que exista fragmentación de cubos para generar las distintas **Franjas del Cubo**, es decir las distintas relaciones en las cuales se dividen los datos de cubo.

Las franjas son similares al concepto de fragmento en el caso de dimensión y representan una división física de la información a almacenar en el cubo.

La diferencia radica en que no se separan atributos sino datos. En general aunque no exista fragmentación de cubos, se puede indicar que existe una sola franja en el cubo, por lo cual se puede decir que el resultado de los pasos 7 al 15 son las distintas franjas del cubo más que el cubo en sí.

La regla aplicada por este paso es la misma que la aplicada en el paso 10.

Nuevamente, [Larr06] propone utilizar el cubo obtenido directamente de la BDF (sin considerar el cubo intermedio) y aplicar sobre este los filtros.

Postcondición

Al terminar esta fase se deben tener:

- El diseño lógico que implementa el modelo CMDM de la fase III
- El diseño de la carga para el modelo antes mencionado desde la fuente canónica
- La matriz de trazabilidad entre conceptos de la realidad, requerimientos y cubos

Matriz de Trazabilidad:

Una vez finalizada la ejecución de este algoritmo, se está en condiciones de crear la matriz de Trazabilidad completa. Para esto sólo basta tomar la Matriz de requerimientos de la Fase 2 de MD4DW [MD06], agregar una columna llamada **Cubos** a la derecha de los requerimientos.

Por cada Cubo, obtener los conceptos que influyen en él y agregar el nombre de dicho cubo en la fila correspondiente al concepto.

Ejemplo:

Concepto	RQ1	RQ2	Cubos
Concepto1		√	Cubo1
Concepto2	√	√	Cubo2

Referencias

[Per01] - V. Peralta. "Diseño Lógico de Data Warehouses a partir de esquemas conceptuales multidimensionales", 2001

[Lar06] - I. Larrañaga. "Carga de un Datawarehouse a partir de la Traza de Diseño", 2006

[MD06] - C. Colombatto, D. Gayoso, S. Giménez. "MD4DW Metadata For DataWarehouse Guide", 2006

[Ber00] - Philip A. Bernstein, Alon Y. Halevy, Rachel a. Pottinger. "A Vision for Management of Complex Model", 2000

[Mar00] - A. Marotta. "Data Warehouse Design and Maintenance through Schema Transformations", 2000

[DSS06] - Germán Asiz, Hermés Piqué. Diseño de un DSS a partir de metadata expresada en OWL. InCo, Facultad de Ingeniería, Universidad de la República del Uruguay - Mayo 2006

[DAML] - Daml.org. <http://www.daml.org/>

[CSI] - <http://www.fing.edu.uy/inco/grupos/csi/>

[Prt] - <http://protege.stanford.edu/>

[Car00] - Carpani, Fernando. CMDM: Un Modelo Conceptual para la Especificación de Bases Multidimensionales. InCo, Facultad de Ingeniería, Universidad de la República del Uruguay. Agosto 2000.

[OWL2] - <http://www.w3.org/TR/owl2-profiles/>

[AIVA] - Empresa cliente en el caso de estudio. www.aiva.com

[Coh99] - D. Cohen. Sistemas de Información Para la Toma de Decisiones.

Ed. Mc Graw Hill, 1999. ISBN 9701008820

<http://www.agapea.com/libros/Sistemas-de-informacion-para-la-toma-de-decisiones-2-Ed--isbn-9701008820-i.htm#otros>

[O2RD06] – E. Vysniauskas, L. Nemuraite. TRANSFORMING ONTOLOGY REPRESENTATION FROM OWL TO RELATIONAL DATABASE. ISSN 1392 – 124X

<http://itc.ktu.lt/itc353/Vysnia353.pdf>

[MJ97] - Matthias Jarke, Yannis Vassiliou: "Data Warehouse Quality: A Review of the DWQ Project"

[RE06] – Ramez Elmasri, Shamkant B. Navathe: "Fundamentals of Database Systems"

[KT96] - Kenan Technologies: "An Introduction to Multidimensional Databases".