

Transformaciones entre Modelo de Base de Datos en el contexto de MDA

Nector Martín Báez
Tesina de grado

Agenda

- ▶ Objetivos
- ▶ Desarrollo dirigido por modelos
- ▶ Model Driven Architecture(MDA)
- ▶ Domain Specific Language(DSL)
- ▶ Eclipse Modeling Project
- ▶ Diseño de Base de Datos y de Datawarehouse
- ▶ MDA y el diseño de datawarehouse
- ▶ *“Aplicando MDA al diseño de un DW temporal”*(propuesta)
- ▶ Toolkit DSL para diseñar un DW temporal(implementación)
- ▶ Demo del toolkit
- ▶ Conclusiones
- ▶ Trabajos Futuros

Objetivos

- ▶ Implementar una propuesta para generar un DW automáticamente partiendo de una visión conceptual de las bases de datos(ER)



- ▶ Analizar mediante esta si MDD es viable para el área de base datos
- ▶ Llevar a cabo este proceso con el soporte de modelado de Eclipse

Desarrollo dirigido por modelos(MDD)

- ▶ Nuevo paradigma de desarrollo software.
 - Idea que surgió en los 70s(Structured Analysis and System Specification. De Marco).
 - Hoy en día se intenta lograr lo que no se pudo con las herramientas CASE que surgieron en los 80s.
- ▶ Se van generando modelos desde los más abstractos a los más concretos.
 - ▶ Los modelos no son solo documentativos ,son la esencia y el código es una consecuencia de los modelos
- ▶ Transformaciones
“Es el proceso de convertir un modelo en otro” (MDA Guide)
- ▶ “La transformación entre modelos constituye el motor del MDD”

MDD – Puntos clave

▶ Abstracción

- Ocultando lo que es irrelevante, un sistema complejo se puede reducir a algo comprensible y manejable.
- Una cualidad importante de los modelos es que son independientes de la plataforma

▶ Automatización

- Capacidad para transformar modelos expresados mediante conceptos de alto nivel, específicos del dominio, en sus equivalentes programas informáticos (transformaciones automáticas)

▶ Estándares

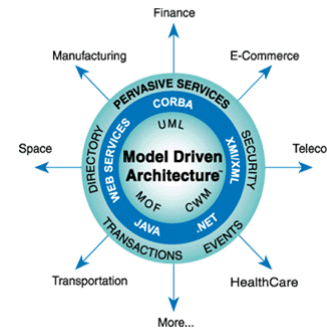
- Estándares industriales abiertos que permiten a los fabricantes de herramientas centrar su atención en su principal área de experiencia.
- Ejemplo Si desarrollo una herramienta de transformaciones no me preocupo por los editores de los modelos que transformo

Model Driven Architecture(MDA)

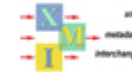
- ▶ La OMG



- ▶ MDA y Su visión general

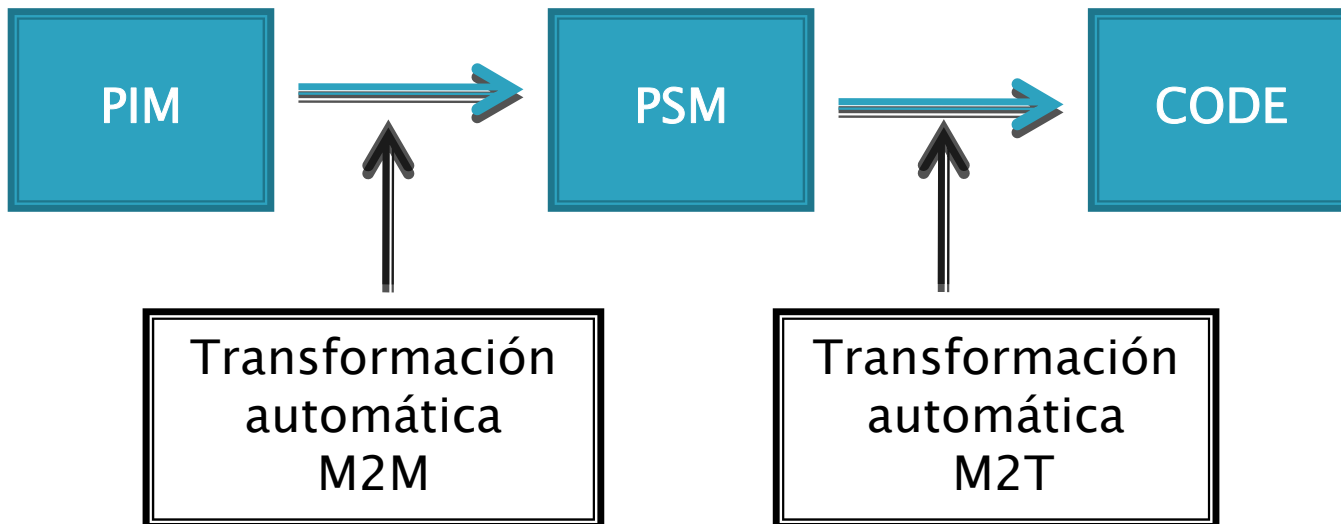


- ▶ Sus estándares



MDA – Visión General

- ▶ Como se debe desarrollar: Modelando
 - ▶ PIM: Modelo Independiente de la plataforma
 - ▶ PSM: Modelo dependiente de la plataforma
 - ▶ Código

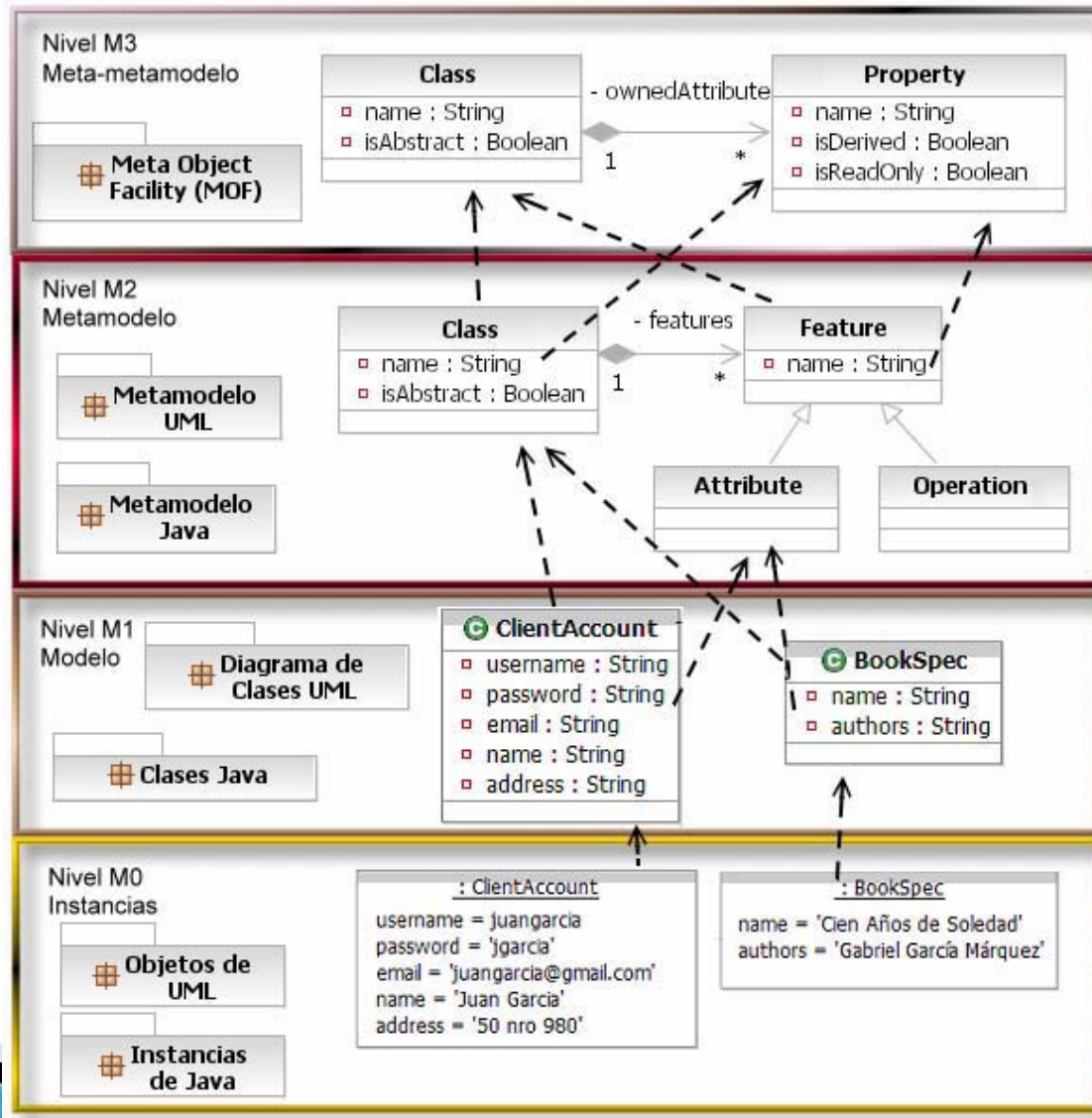


MDA – Sus 4 capas de abstracción

- ▶ M3 Meta–Meta–Modelado
 - Se define el lenguaje para representar un lenguaje de metamodelado.
- ▶ M2 Meta–Modelado
 - En este nivel se definen los artefactos que utilizamos para modelar(Clase, atributo, jerarquía, etc)
- ▶ M1 Modelado
 - El modelo del sistema. Representan las categorías de las instancias(ej UML, perfiles, etc)
- ▶ Instancias
 - Instancias reales del sistema, los objetos de la aplicación
- ▶ Lenguajes de transformacion
- ▶ XMI para persistir modelos



MDA - Ejemplo

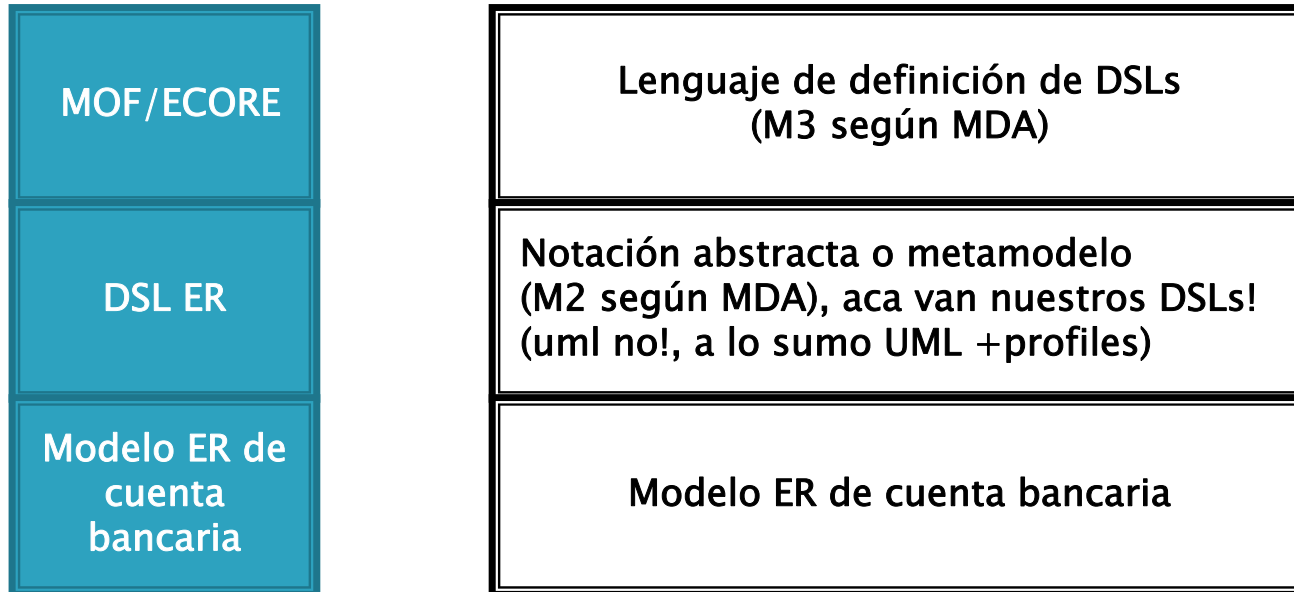


Domain Specific Language(DSL)

- ▶ Motivación:
 - “General to general mapping typically don´t work well” (UML a Java por ejemplo)(EMP)
 - Acercar los modelos a los expertos de cada dominio
- ▶ No existe una definición precisa:
 - Según Eclipse Modeling Project: “Lenguaje diseñado para ser útil en una tarea específica”
- ▶ Ejemplos: BNF, SQL, LATEX
- ▶ Ejemplo muy emblemático: UML especializado con perfiles.(Ej: perfil J2EE)
 - Casos: AndroMDA, ArcStyler y muchos mas

Eclipse Modeling Project

- ▶ Abstracción y formalización con DSL/EMP(similar a MDA)



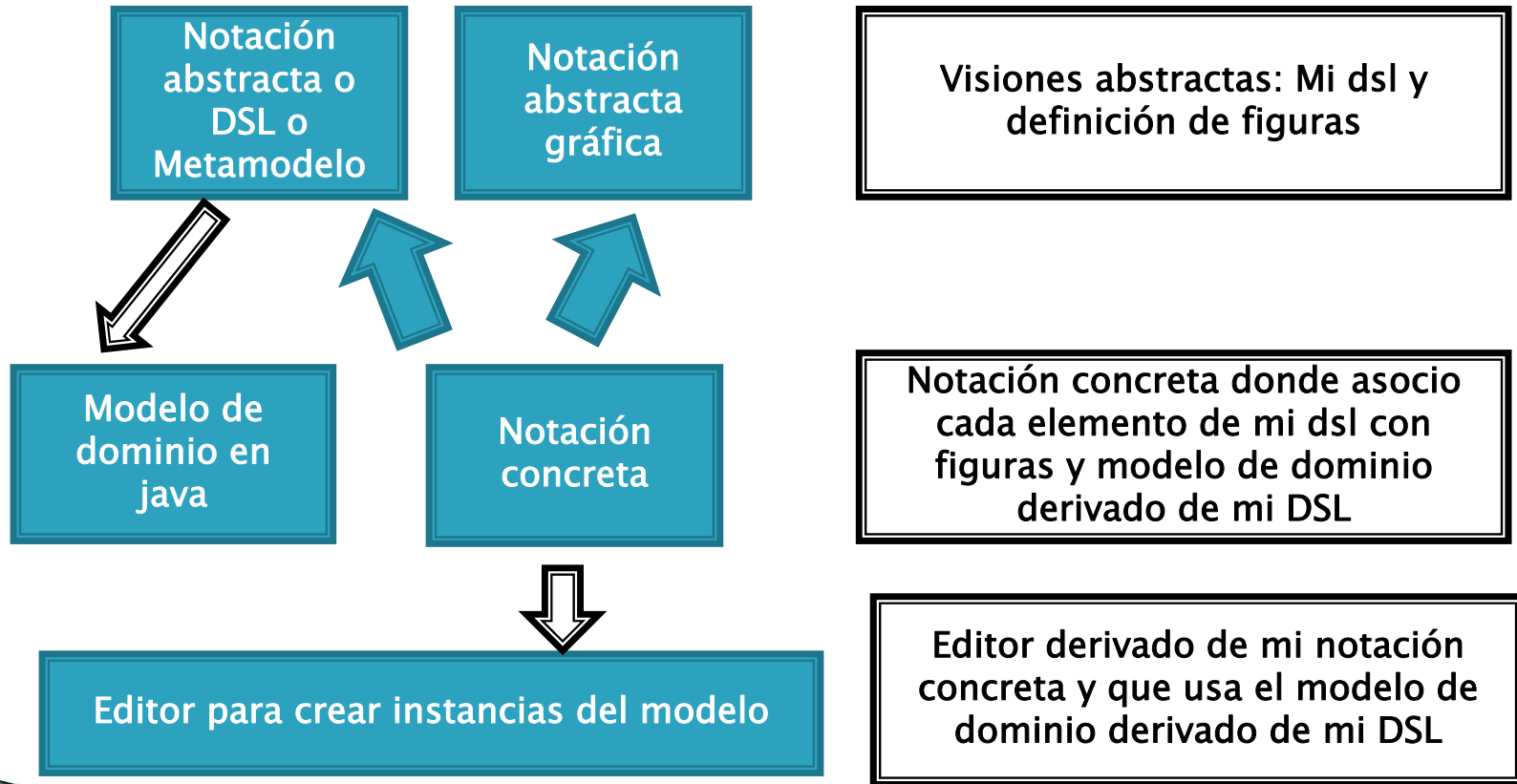
- ▶ Abstracción: No usa el concepto de metamodelo sino de notación abstracta
- ▶ Afirman que los términos modelo, meta-modelo y meta-meta modelos confunden bastante

Eclipse Modeling Project

- ▶ Colección de proyectos y tecnologías MDD
- ▶ Soporte para la creación de DSLs
- ▶ Implementa muchos estándares de la OMG
- ▶ MDA con EMP es una opción(EMP no usan para nada los términos de PIM y PSM)
- ▶ La mención a MDA de EMP es casi nula
- ▶ Organizado en subproyectos que dan soporte para
 - Sintaxis abstracta: EMF(ecore), VF
 - Sintaxis concreta: GMF, TMF(Xtext, TCS)
 - Transformaciones
 - Model to Model(M2M): ATL, QVT
 - Model to Text(M2T): JET, Xpand

Eclipse Modeling Project

- ▶ Soporte para DSLs con EMP(editores para nuestros DSLs)



Diseño de Base de Datos

- ▶ Utilizado para diseñar como almacenamos datos.
 - Abstracción: Se utilizan modelos para diseñar.
 - Automatización: de manera formal e informal(no automática) pasamos de un modelo a otro.
 - Estándares: Estos modelos se utilizan desde hace muchos años. Se han convertido en estándares de facto.
- ▶ Existe aquí algo de lo que se busca con MDD

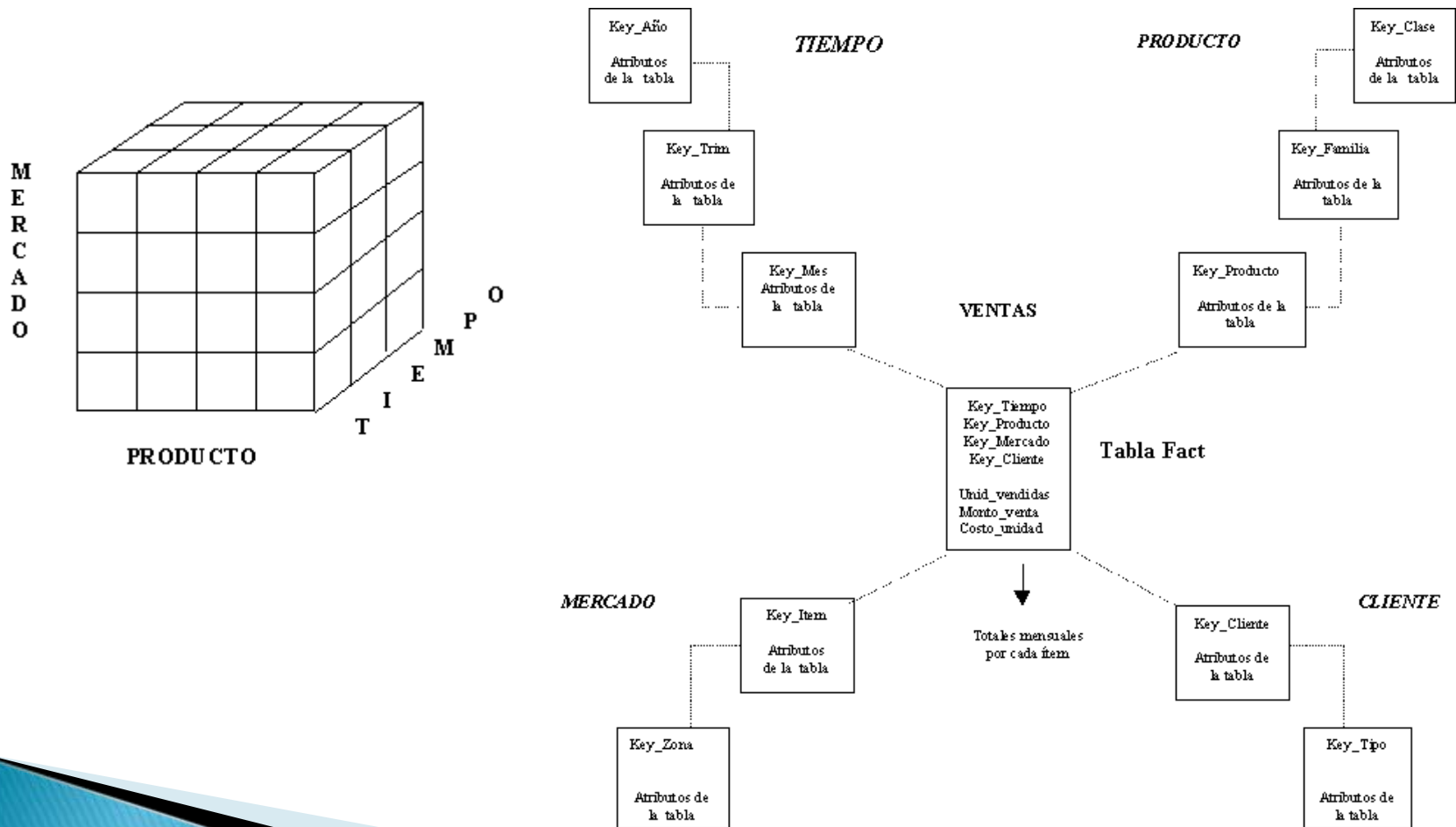
Diseño de Base de Datos

- ▶ Modelos Independiente de la plataforma(Diseño conceptual)
 - Abstracción de la realidad
- ▶ Modelos independientes de la tecnología(Diseño Lógico)
 - Modelo lógico de tablas relacionadas por atributos
- ▶ Código(Dependiente de cada RDBMS)
 - ANSI SQL,

Tenemos abstracción, transformaciones(informales) y estándares (de facto) algo parecido a lo que se busca con MDD

Datawarehouse

- ▶ Colección de datos orientadas a un ámbito



Datawarehousing

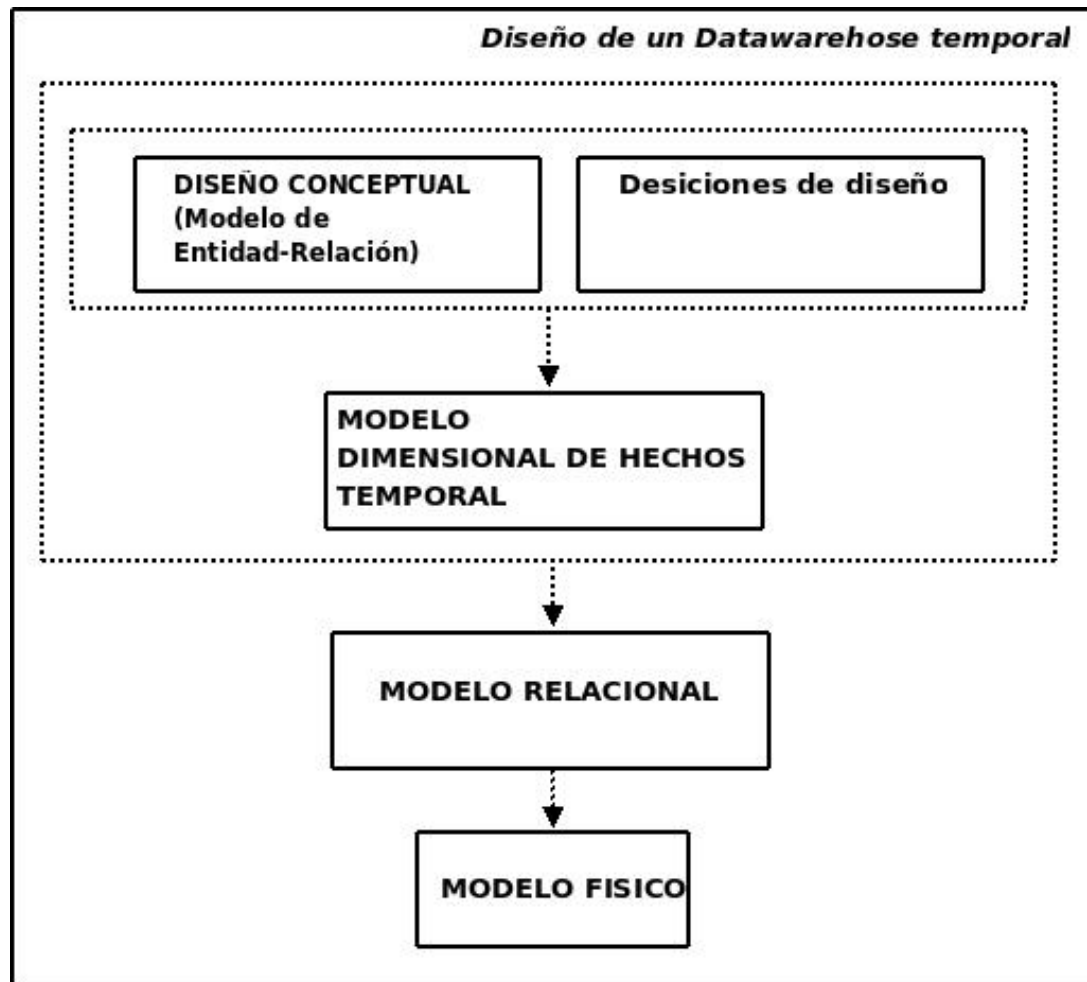
- ▶ Definiciones aceptadas:
 - *“Una copia de las transacciones de datos específicamente estructurada para la consulta y el análisis” (Ralph Kimball)*
 - *En términos de las características del repositorio de datos(Bill Immon):*
 - *Orientado a temas, Variante en el tiempo, No volátil, Integrado*
- ▶ Modelados mediante el Modelo multidimensional(alternativa al modelo ER)
- ▶ Contiene
 - Hechos: Un evento o hecho de interés para la toma de decisiones
 - Dimensiones: Diferentes perspectivas mediante las que analizamos los hechos
 - Jerarquías: Las jerarquías representan los diferentes niveles de sumariazación de la información.
 - Estrictas y No Estrictas
- ▶ Existen varias propuestas para su diseño
 - Intentamos comenzar a trabajar a nivel conceptual(PIM)

Datawarehousing

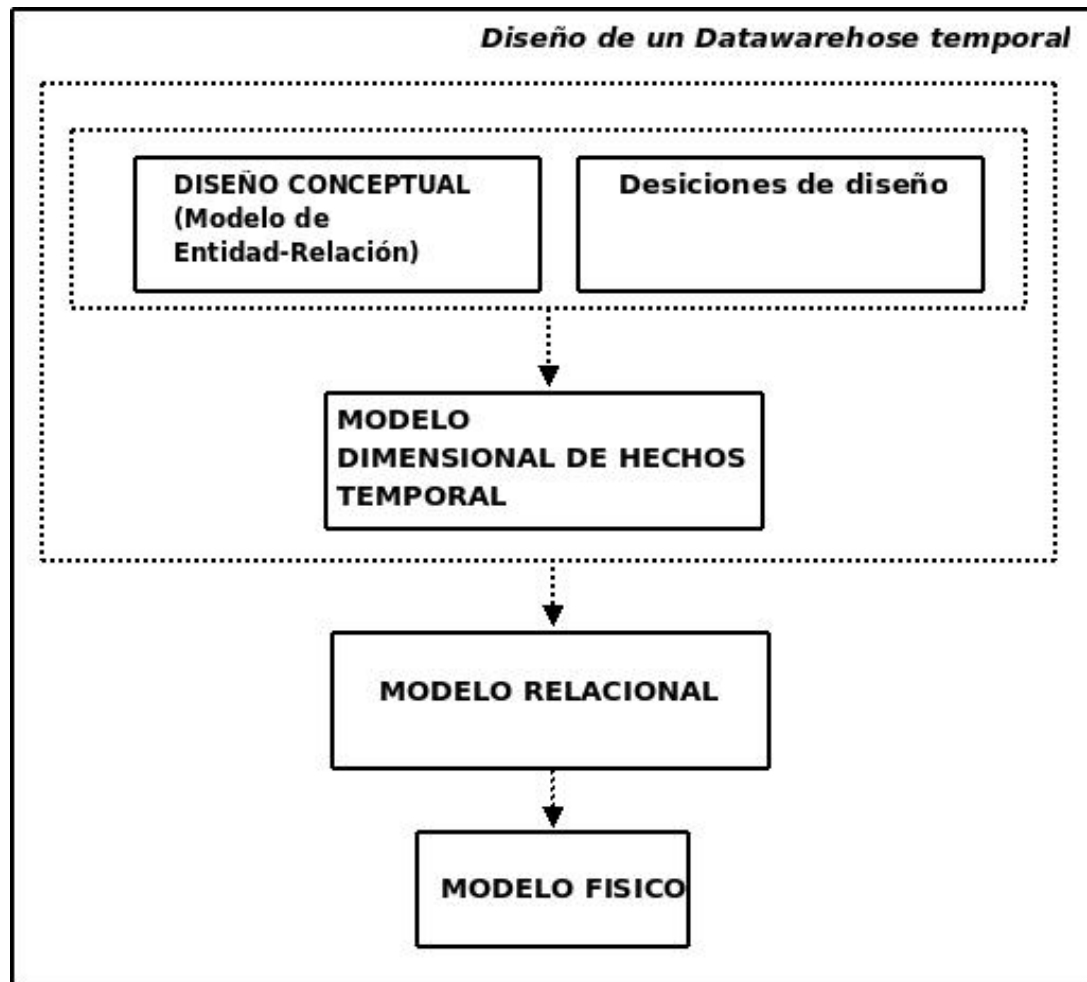
- ▶ Definiciones aceptadas:
 - *“Una copia de las transacciones de datos específicamente estructurada para la consulta y el análisis” (Ralph Kimball)*
 - *En términos de las características del repositorio de datos(Bill Immon):*
 - *Orientado a temas, Variante en el tiempo, No volátil, Integrado*
- ▶ Modelados mediante el Modelo multidimensional(alternativa al modelo ER)
- ▶ Contiene
 - Hechos: Un evento o hecho de interés para la toma de decisiones
 - Dimensiones: Diferentes perspectivas mediante las que analizamos los hechos
 - Jerarquías:
 - Estrictas y No Estrictas
- ▶ Existen varias propuestas para su diseño
 - Intentamos comenzar a trabajar a nivel conceptual(PIM)

Nuestro objetivo: *“Implementar una propuesta para diseñar DWs a partir de BD operacionales partiendo de una visión conceptual de las mismas”*

Diseño de Datawarehousing

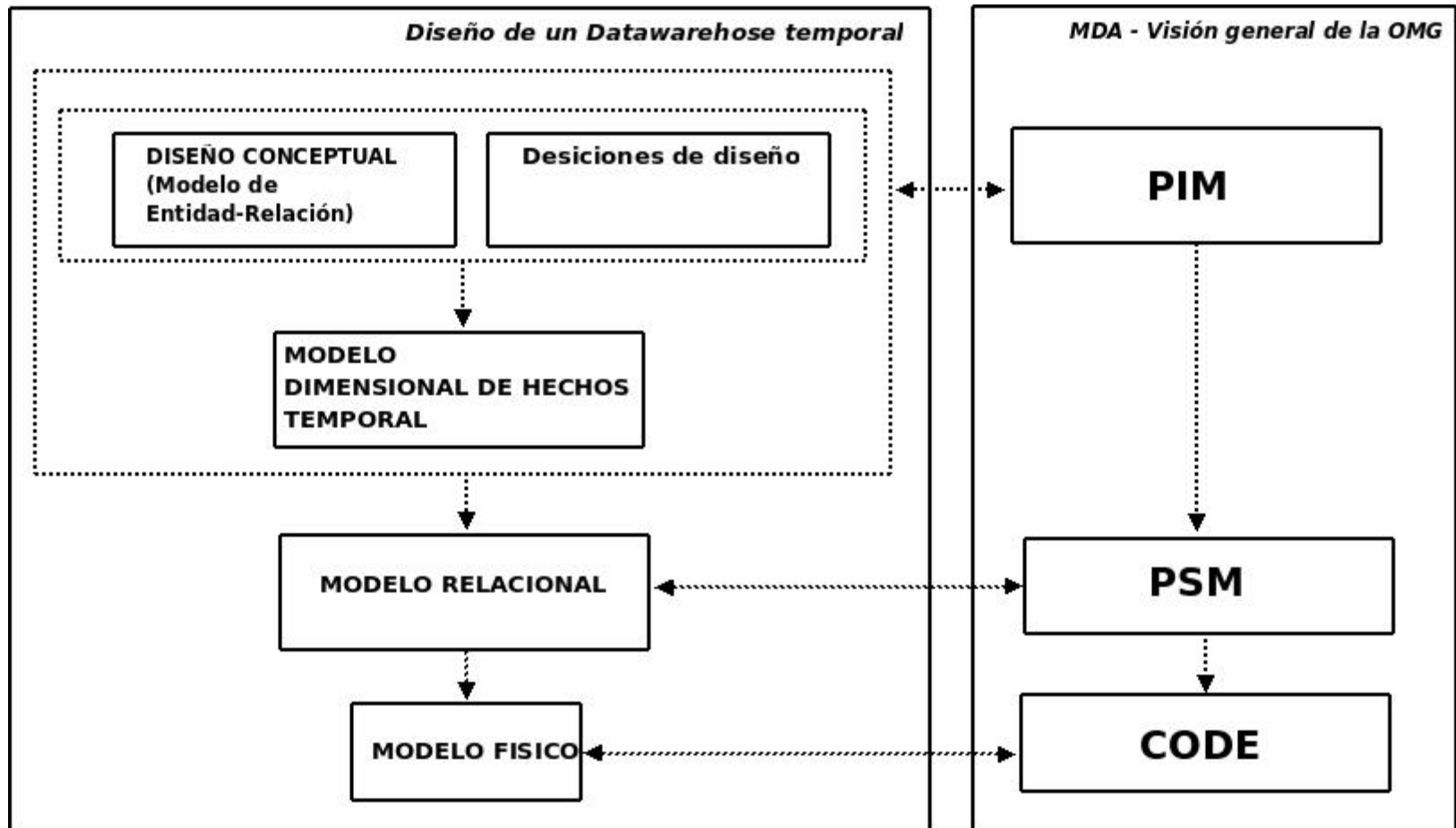


Diseño de Datawarehousing

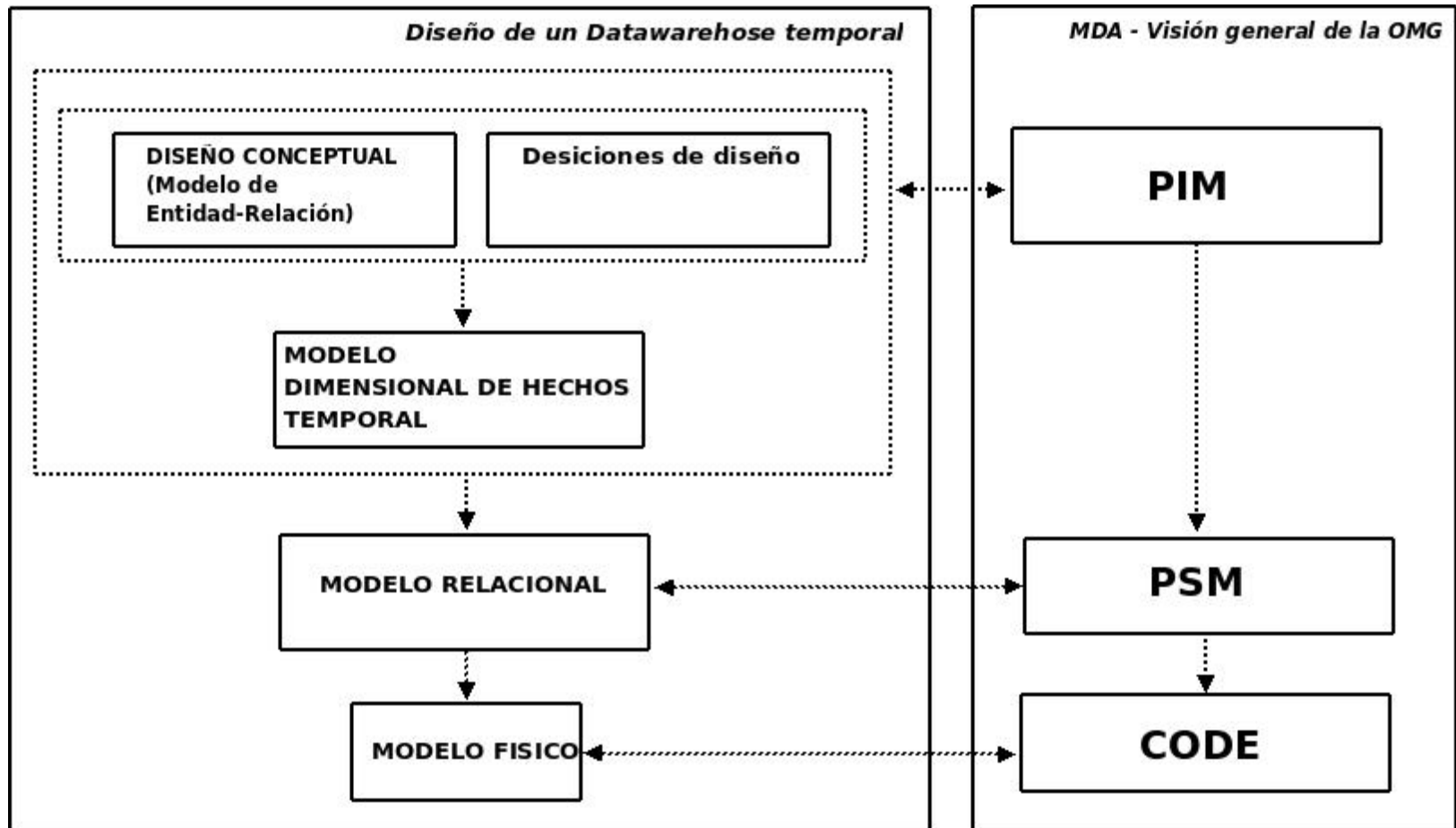


¿A que se parece?

Diseño Datawarehousing con MDA



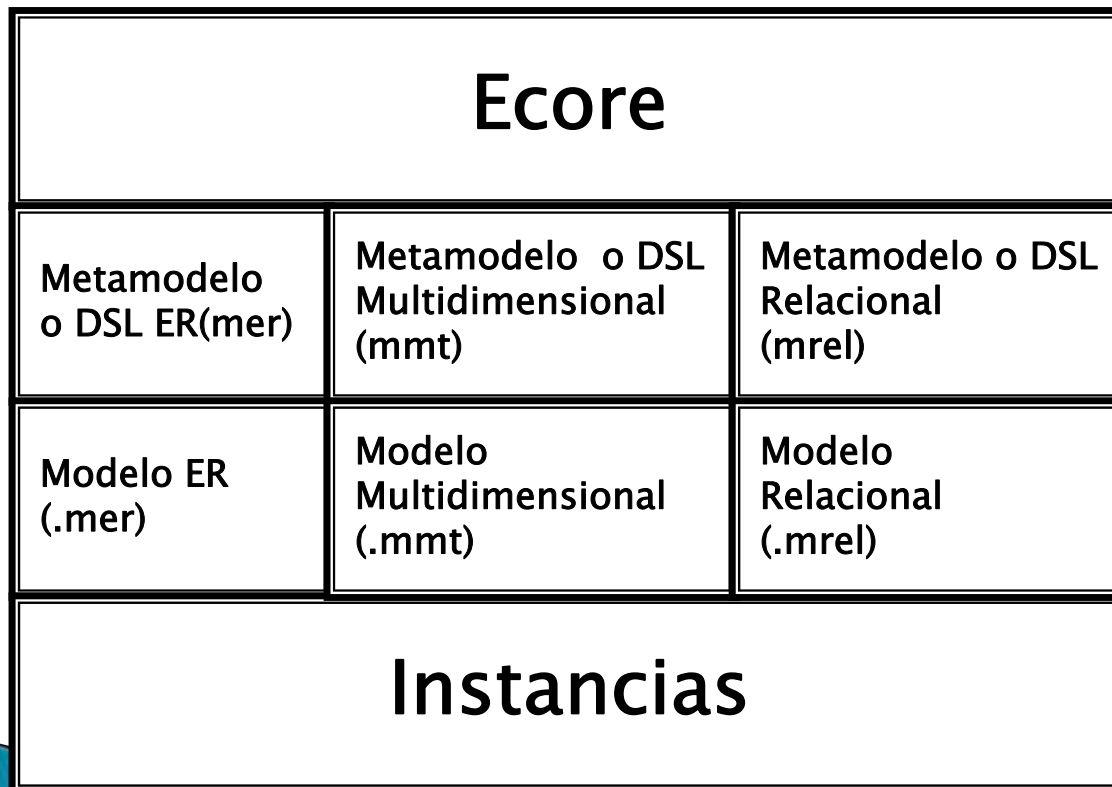
Diseño Datawarehousing con MDA



Nuestro objetivo: *“Implementar cada modelo como un dsl y las transformaciones entre los mismos.”*

Diseño Datawarehousing con MDA

- Estándarizamos:
 - Llenamos el hueco en M2(MDA)
 - Notaciones abstractas(creamos DSLs)



Lenguaje de modelado de EMP en respuesta a MOF de la OMG

Nuestros metamodelos para estandarizar los modelos utilizados en el diseño de base de datos

Los modelos que utilizamos para diseñar bds

Las instancias de los modelos que implementamos

Aplicando MDA al diseño de un DW temporal

- ▶ Propuesta existente en la que nos basamos
 - Se intenta diseñar el DW conceptualmente mediante el Modelo Hecho Dimensional
- ▶ Pasos:
 1. Partimos de un modelo ER
 2. Especificamos hechos de interés
 3. Obtenemos el ER reestructurado
 4. Obtenemos un Grafo de atributos
 5. Obtenemos un Modelo multidimensional (Modelo hecho dimensional)
 6. Obtenemos un Modelo Relacional
 7. Derivamos Código SQL DDL

Aplicando MDA al diseño de un DW temporal

- ▶ Propuesta existente en la que nos basamos
 - Se intenta diseñar el DW conceptualmente mediante el Modelo Hecho Dimensional
 - Se le agrega temporalidad(tiempo válido)
- ▶ Pasos:
 1. Partimos de un modelo ER
 2. Especificamos el hecho de interés
 3. Obtenemos el ER reestructurado
 4. Obtenemos un Grafo de atributos
 5. Obtenemos un Modelo multidimensional(Modelo hecho dimensional)
 6. Obtenemos un Modelo Relacional
 7. Derivamos Código SQL DDL

Nuestro objetivo: “*Implementar esta propuesta*” → DwTk

DwTk – Editores de los modelos

- platform:/resource/ar.edu.unlp.info.ba
 - mrel
 - Schema
 - Table -> Named
 - Column -> Named
 - KeyColumn -> Column
 - ForeignKeyColumn -> Column
 - DataColumn -> Column
 - Named
 - KeyRelationship
 - Cardinality
 - ColumnType
 - REFERENCE_OPTIONS

Notaciones abstractas en ecore

DwTk – Editores de los modelos

- platform:/resource/ar.edu.unlp.info.ba
- mrel
 - Schema
 - Table -> Named
 - Column -> Named
 - KeyColumn -> Column
 - ForeignKeyColumn -> Column
 - DataColumn -> Column
 - Named
 - KeyRelationship
 - Cardinality
 - ColumnType
 - REFERENCE_OPTIONS

Mediante notaciones concretas de esta derivamos nuestro modelo de dominio y nuestro editor

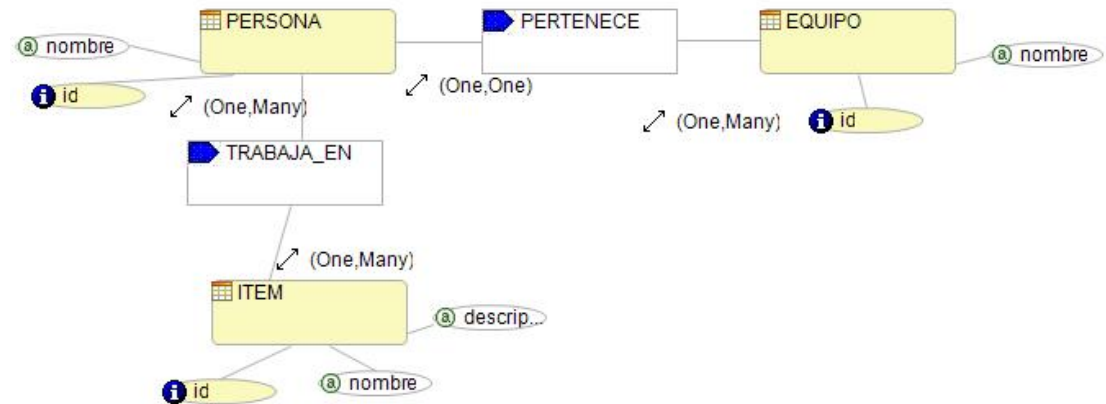
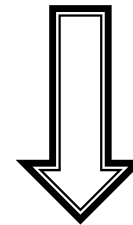
Notaciones abstractas en ecore

DwTk – Editores de los modelos

- platform:/resource/ar.edu.unlp.info.ba
- mrel
 - Schema
 - Table -> Named
 - Column -> Named
 - KeyColumn -> Column
 - ForeignKeyColumn -> Column
 - DataColumn -> Column
 - Named
 - KeyRelationship
 - Cardinality
 - ColumnType
 - REFERENCE_OPTIONS

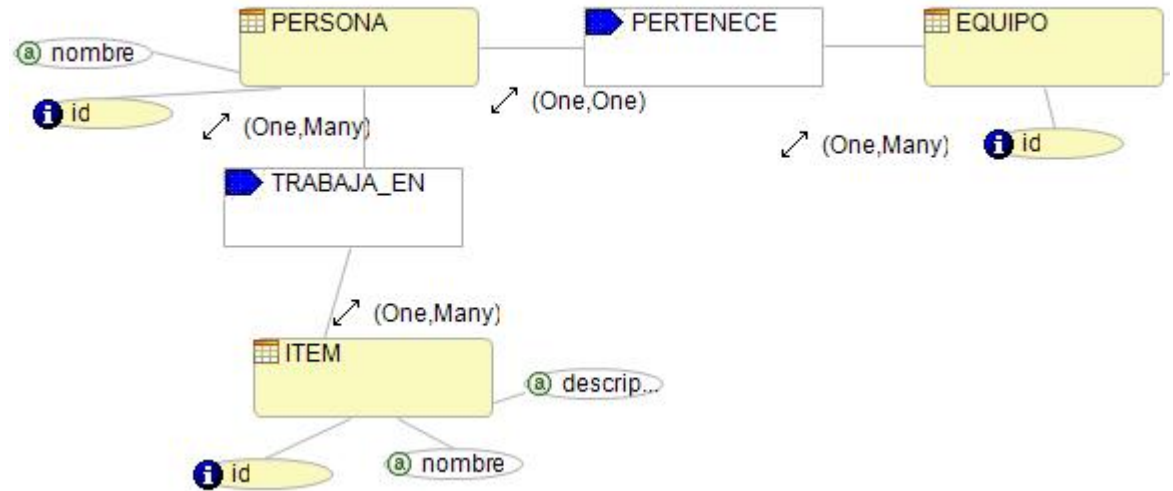
Mediante notaciones concretas de esta derivamos nuestro modelo de dominio y nuestro editor

Notaciones abstractas en.ecore



DwTk - Modelo ER(PIM)

- amds
 - ERSchema
 - Named
 - Entity -> RelationalElement
 - Relationship -> RelationalElement
 - RelationshipEnd
 - Attribute -> Named
 - RelationalElement -> Named
 - MultiplicityType
 - Zero = 0
 - One = 1
 - Many = 2
 - Type
 - AttributeType
 - AttributeElementAsociation
 - IdentifyAttribute -> Attribute
 - DescriptiveAttribute -> Attribute
 - WeakEntity -> RelationalElement
 - ForeignIdentifier



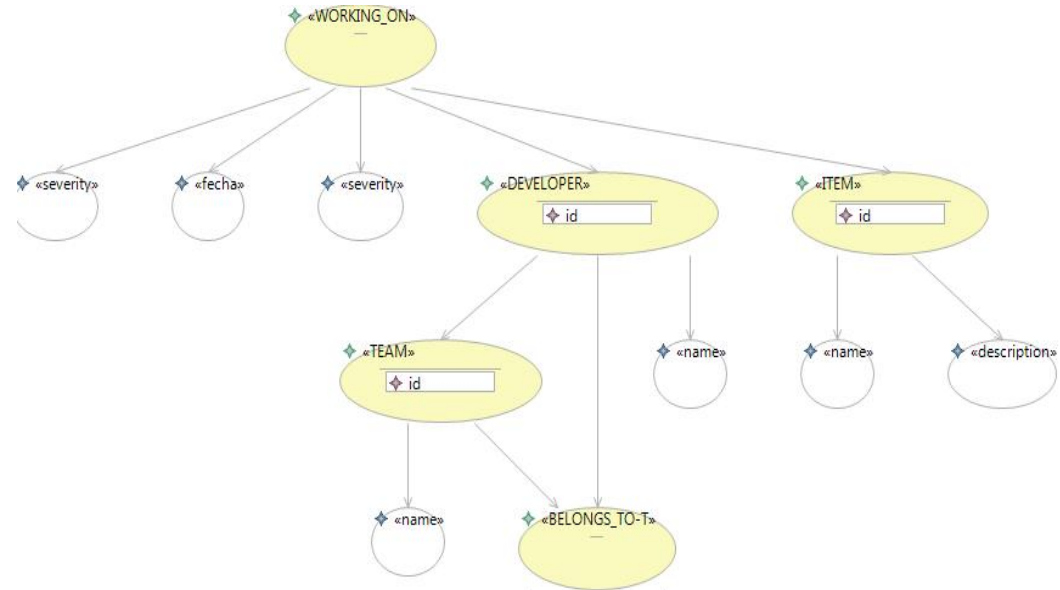
Notación abstracta de Entidad-Relación

Editor del Modelo de Entidad-Relación

DwTk – Grafo de Atributos(PIM)

- platform/resource/at.edu.ump
 - graph
 - Schema
 - Named
 - Vertex -> Named
 - Leaf -> Vertex
 - Node -> Vertex
 - MainNode -> Vertex
 - Identifier -> Named
 - AttributeType

Notación abstracta del grafo

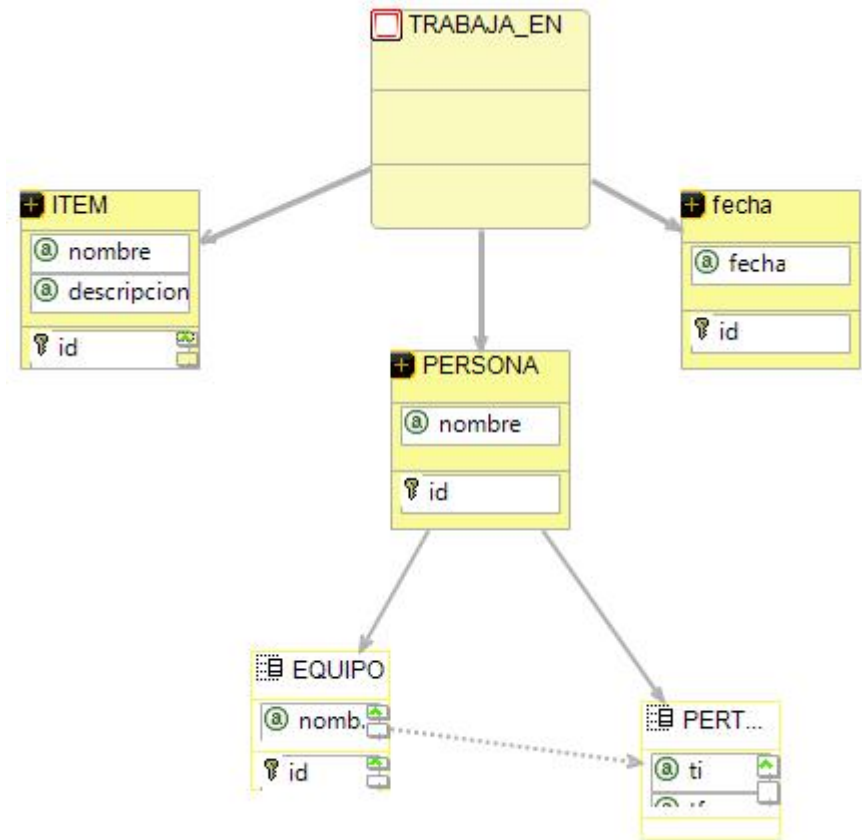


Editor del grafo de atributos

DwTk-Modelo Multidimensional(PIM)

- mmt
 - Schema
 - Named
 - MultidimensionalElement -> Named
 - Fact -> MultidimensionalElement
 - Dimension -> MultidimensionalElement
 - Hierarchy -> MultidimensionalElement
 - Attribute -> Named
 - Identifier -> Named
 - AttributeType

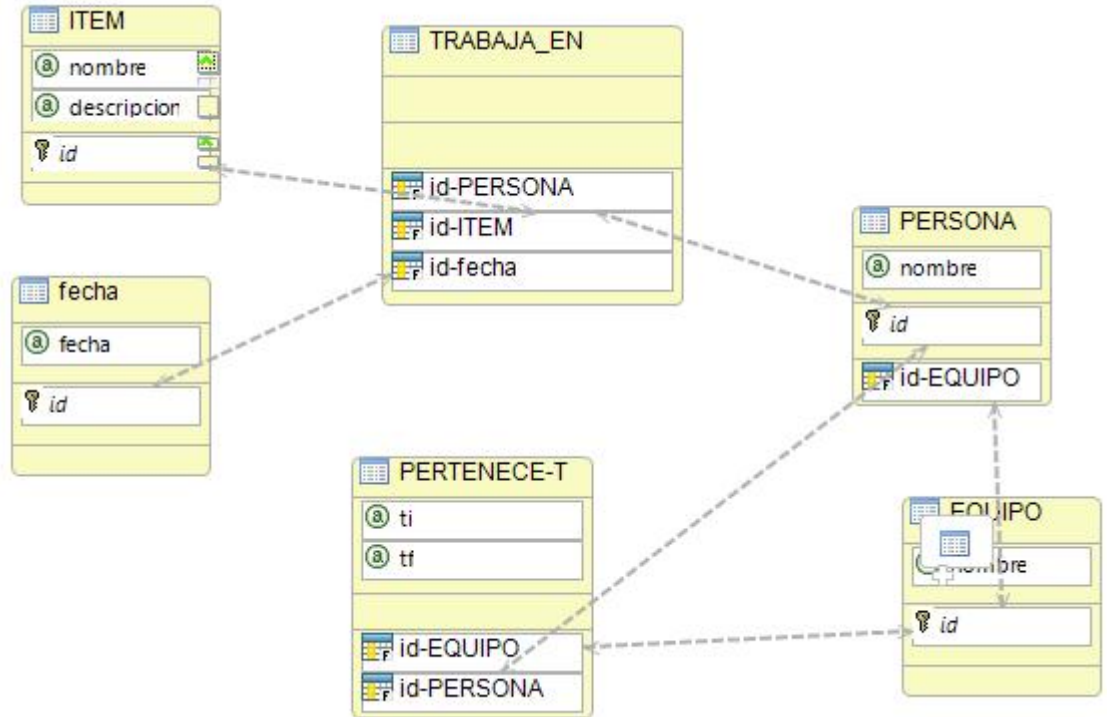
Notación abstracta
multidimensional en ecore



Editor multidimensional

DwTk – Modelo Relacional(PSM)

- ✚ mrel
 - ▷ Schema
 - ▷ Table -> Named
 - ▷ Column -> Named
 - ▷ KeyColumn -> Column
 - ▷ ForeignKeyColumn -> Column
 - ▷ DataColumn -> Column
 - ▷ Named
 - ▷ KeyRelationship
 - ▷ Cardinality
 - ▷ ColumnType
 - ▷ REFERENCE_OPTIONS



Notación abstracta
relacional en ecore

Editor Relacional

DwTk – Transformaciones

▶ M2M

- Llevadas a cabo mediante el lenguaje ATL
- ATL convierte modelos en ecore(y UML2 eventualmente)
- Tuvimos que hacer un patch para la versión 2 de ATL que no estaba preparada para ser usada “por fuera” del workspace de Eclipse. Esto fue corregido en la versión 3 de ATL cuando toma forma la idea de “Eclipse as a DSL Toolkit” 😊

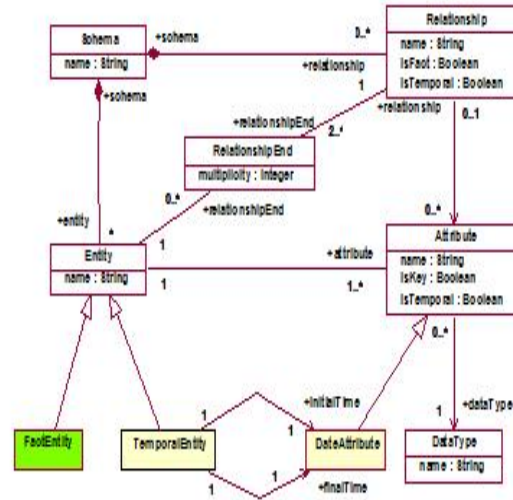
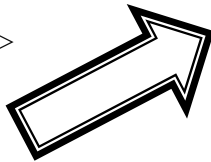
▶ M2T

- Implementamos la derivación de código a mysql(restan corregir detalles)

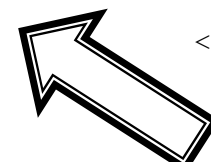
|

Dwtk – Reestructuración del modelo ER

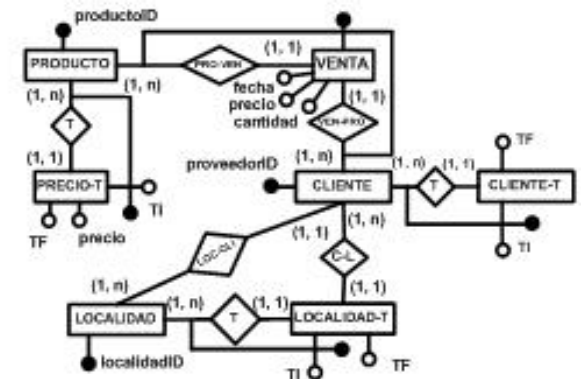
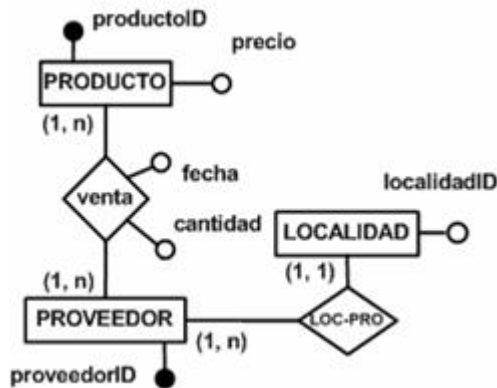
<<CONFORMS>>



<<CONFORMS>>



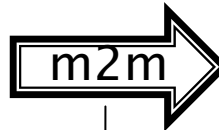
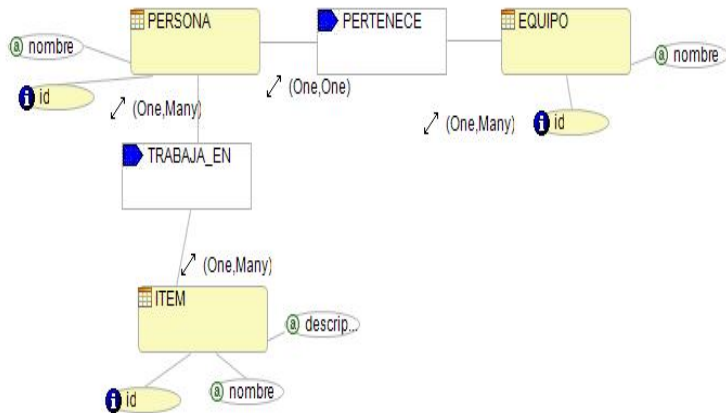
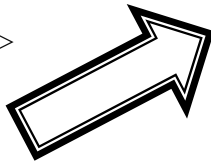
m2m



Dwtk – Reestructuración del modelo ER

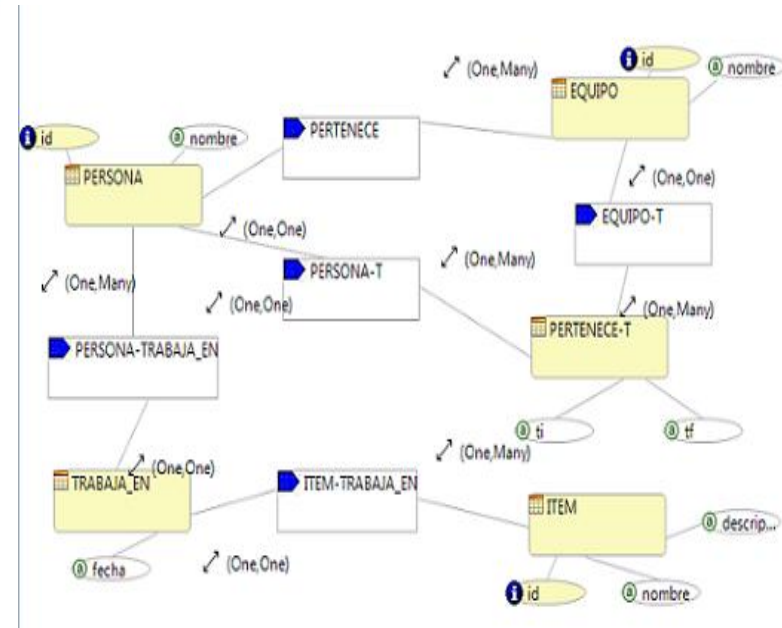
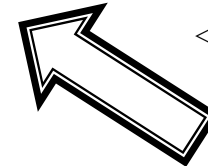
- amds
 - ERSchema
 - Named
 - Entity -> RelationalElement
 - Relationship -> RelationalElement
 - RelationshipEnd
 - Attribute -> Named
 - RelationalElement -> Named
 - MultiplicityType
 - Zero = 0
 - One = 1
 - Many = 2
 - Type
 - AttributeType
 - AttributeElementAsociation
 - IdentifyAttribute -> Attribute
 - DescriptiveAttribute -> Attribute
 - WeakEntity -> RelationalElement
 - ForeignIdentifier

<<CONFORMS>>



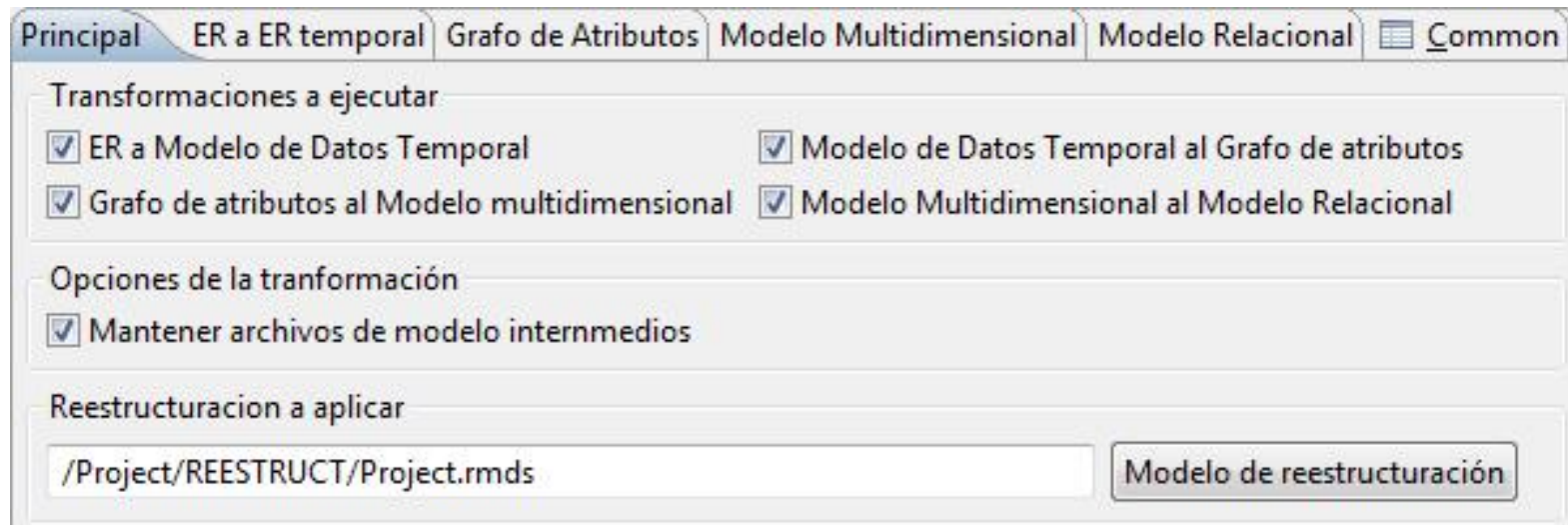
Declare mainFact TRABAJA_EN
Declare temporal PERTENECE

<<CONFORMS>>



Dwtk – Transformaciones implementadas


1. Modelo ER Reestructurado al Grafo de atributos
2. Grafo al Modelo Multidimensional temporal
3. Modelo Multidimensional al Relacional
4. Modelo Relacional a Código



The screenshot shows the Dwtk software interface with the following components:

- Navigation Tabs:** Principal (selected), ER a ER temporal, Grafo de Atributos, Modelo Multidimensional, Modelo Relacional, and Common.
- Transformaciones a ejecutar:** A section containing four checked checkboxes:
 - ER a Modelo de Datos Temporal
 - Modelo de Datos Temporal al Grafo de atributos
 - Grafo de atributos al Modelo multidimensional
 - Modelo Multidimensional al Modelo Relacional
- Opciones de la transformación:** A section containing one checked checkbox:
 - Mantener archivos de modelo intermedios
- Reestructuración a aplicar:** A section containing a text input field with the path `/Project/REESTRUCT/Project.rmds` and a button labeled "Modelo de reestructuración".

Dwtk – Generación automática de código

- ▶ Transformamos el modelo relacional en scripts de sql ddl(Create)
 - ▶ No hay estándar para Sql ddl
 - ▶ Implementamos solo sql ddl para mysql a modo de ejemplo
 - ▶ Utilizamos el lenguaje Xpand(no es estándar de la OMG)
- 

Dwtk - Demo

- ▶ Ejemplo: Desarrolladores de un proyecto y los items en los que estos trabajan.
- ▶ Derivaremos del modelo conceptual ER un DW y luego generaremos código SQL ddl

Conclusiones

- ▶ Se cumplió el objetivo de implementar la propuesta MDA tomada como referencia usando estándares de la OMG construyendo DSLs
- ▶ Concluimos que es mejor usar MMD(MDA y/o DSLs) en aquellos dominios donde los modelos ya se usan y son estándares defacto
- ▶ Mostramos en particular la aplicabilidad del desarrollo dirigido por modelos para diseñar base de datos y datawarehousing
- ▶ Sobre el estándar CWM de la OMG

Trabajos relacionados y futuros

▶ Trabajos relacionados

- El trabajo “Usando ATL en Transformación de Modelos Multidimensionales Temporales”(soy coautor y ahí probamos y decidimos usar ATL)
- Aplicando MDA al Diseño de un Datawarehouse Temporal(es nuestra guía)

▶ Trabajos futuros

- Compatibilidad con CWM
- Implementación de aspectos del diseño de base de datos y datawarehousing no considerados
- Derivación automática de consultas a partir del grafo
- Derivación automática de ETL
- Generalizar algunas cosas de nuestro toolkit(darle mas forma de framework)

FIN

- ▶ ¿Preguntas?

FIN

- ▶ Muchas gracias!