# Un método global basado en UML para el diseño de Almacenes de Datos*

Sergio Luján-Mora and Juan Trujillo

Dept. de Lenguajes y Sistemas Informáticos
Universidad de Alicante (España)
{slujan,jtrujillo}@dlsi.ua.es

**Resumen** Un Almacén de Datos (*Data Warehouse*, DW) es un sistema de información complejo utilizado principalmente en el proceso de toma de decisiones mediante el uso de las aplicaciones de Procesamiento Analítico en Línea (*On-Line Analytical Processing*, OLAP). Hasta el momento, se han propuesto varias aproximaciones para acometer el diseño de las distintas partes de un DW como pueden ser los esquemas conceptual y lógico del DW o los procesos de Extracción, Transformación y Carga de datos (*Extraction-Transformation-Loading*, ETL). Sin embargo, en la actualidad no existe un método ampliamente aceptado para realizar el diseño de un DW que abarque todas sus fases e integre los distintos modelos utilizados en cada una de ellas de una forma coherente. En este trabajo, presentamos un método global para acometer el diseño de todas las fases y aspectos relevantes de los DW, incluyendo las fuentes de datos operacionales, los procesos ETL y el propio esquema del DW. Las principales ventajas de nuestra propuesta son: el uso de una notación estándar (UML) en los modelos utilizados en todas las fases de diseño, la integración de las distintas fases de diseño en un modelo simple y coherente y el uso de un mecanismo de agrupamiento (los paquetes de UML) que permite al diseñador estructurar los modelos en diferentes niveles de detalle.

**Palabras clave**: almacenes de datos, modelado multidimensional, métodos de diseño, UML

## 1. Introducción

A principios de los noventa, Inmon [11] definió el término "almacén de datos" (*Data Warehouse*, DW): "Un almacén de datos es una colección de datos orientados por temas, integrados, variables en el tiempo y no volátiles para el apoyo de la toma de decisiones". Un DW es "integrado" porque los datos que se introducen en el DW se obtienen de una variedad de fuentes de datos (sistemas heredados, bases de datos relacionales, ficheros COBOL, etc.). Los procesos ETL

* Realizado con una ayuda concedida por la Secretaría de Estado de Educación y Universidades (Ministerio de Educación, Cultura y Deporte).

(*Extraction-Transformation-Loading*) son los responsables de la extracción de los datos a partir de las diversas fuentes de datos heterogéneas, su transformación (conversión, limpieza, etc.) y su carga en el DW.

Por otro lado, es ampliamente aceptado que los DW, las bases de datos multidimensionales (MD) y las aplicaciones OLAP están basadas en el modelado MD. En estos últimos años, se han propuesto varias aproximaciones para acometer el diseño conceptual y lógico de los DW y los sistemas MD, tales como [3,9,21,24,10,23,2], para representar las principales propiedades estructurales y dinámicas del modelado MD. Sin embargo, ninguna de estas propuestas ha sido ampliamente aceptada como un modelo conceptual estándar para acometer el modelado MD. Debido a restricciones de espacio, aconsejamos la lectura de [1] para una comparación y discusión detallada sobre la mayoría de estos modelos.

Sin embargo, dada la gran variedad de modelos utilizados en las distintas fases del diseño de los DW, se hace absolutamente necesario desarrollar un método lo más estándar posible que proporcione guías de diseño para crear y transformar estos modelos durante la fase de desarrollo de un DW. En la actualidad, existen algunos intentos de proporcionar un método para el desarrollo de DW. Sin embargo, desde nuestro punto de vista, ninguno de ellos cubre todas las fases y transformaciones necesarias para disponer de un método global y estándar para el diseño de DW.

En trabajos previos, hemos presentado una propuesta para acometer el diseño conceptual de los DW mediante UML [23,15,16] y obtener de forma automática el esquema lógico correspondiente según la plataforma destino (relacional, MD, etc.). Por otro lado, también hemos realizado una propuesta para realizar el modelado conceptual de los procesos ETL mediante UML [22].

En este trabajo, presentamos un método global orientado a objetos (OO) que integra todas las fases de diseño de los DW desde las fuentes de datos operacionales hasta la implementación, incluyendo la definición de los procesos ETL y los requisitos de usuario. Nuestro objetivo es combinar, en un conjunto de modelos relacionados, todo el análisis y diseño de los DW desde las fuentes de datos operacionales hasta su implementación final. En nuestra aproximación, un modelo de DW está basado en la típica arquitectura de un sistema de DW [12] y se compone de cuatro esquemas y las correspondientes transformaciones entre ellos. Al contrario que otros métodos y propuestas para el modelado MD y diseño de DW, nuestro método es independiente de cualquier implementación específica (relacional, MD, OO, etc.). Finalmente, nuestro método está basado en un lenguaje de modelado estándar (UML), lo que evita que los diseñadores aprendan una nueva notación o lenguaje específicos para el diseño de DW.

El resto del artículo está estructurado como sigue. En la sección 2 se presenta un breve resumen de los métodos más relevantes presentados hasta el momento para acometer el diseño de los DW. En la sección 3 presentamos el esquema general de nuestra propuesta junto con los distintos modelos y mapeos que lo componen. En la sección 4 presentamos los principales pasos de diseño que propone nuestro método. Por último, la sección 5 presenta las principales conclusiones y trabajos futuros extraídos del presente trabajo.

## 2. Estado de la cuestión

En esta sección, presentamos un breve resumen de los métodos más relevantes para el diseño de los DW. Algunos otros métodos que no se pueden comentar por falta de espacio son [8,7,6].

En [13], se presentan varios casos de estudio significativos de *data marts* (DW departamentales) a los que se aplica el esquema estrella (*star schema*) y sus variantes de copo de nieve (*snowflake*) y constelación de hechos (*fact constellation*) para realizar el modelado MD. Además, propone lo que denomina una matriz de arquitectura de BUS para integrar el diseño de varios *data marts* y conseguir así un DW corporativo y global. Si bien consideramos estos trabajos como un referente fundamental en el modelado MD, a nuestro juicio todos ellos adolecen de un método formal para el diseño de DW. En [14], se presenta el ciclo de vida de un proyecto de DW, para el cual los autores proponen distintas herramientas y técnicas a seguir, sin que se proponga un método o modelo global y formal a lo largo de todo el proceso.

En [9], los autores proponen el *Dimensional-Fact Model* (DFM) como una notación propia para acometer el modelado conceptual de los DW. Además, proponen un marco metodológico para definir un esquema conceptual a partir de los esquemas Entidad-Relación (ER) que representan las fuentes de datos operacionales. Analizando los aspectos puramente metodológicos, esta propuesta está únicamente enfocada al diseño conceptual y lógico del DW, ya que no considera un aspecto tan relevante como es el diseño de los procesos ETL. Además, los autores presuponen una implementación relacional del DW y que se dispondrá de los esquemas ER de todas las fuentes de datos operacionales, lo que por desgracia no sucede en muchas ocasiones.

En [3], los autores presentan el *Multidimensional Model* (MD), un modelo lógico para acometer el diseño de DW y un método para construirlo a partir de los esquemas ER de las fuentes de datos operacionales. Aunque los pasos del método están definidos en una forma coherente y lógica, el diseño del DW está basado únicamente en las fuentes de datos operacionales, lo que a nuestro juicio es insuficiente puesto que estos sistemas son eminentemente de consulta, por lo que también se tiene que tener en cuenta en su diseño los requisitos de usuario.

En [18] se propone de nuevo cómo construir el esquema estrella (y sus diferentes variantes) a partir de los esquemas conceptuales de las fuentes de datos operacionales de que disponga la empresa. Una vez más, presupone que las fuentes de datos están definidas mediante esquemas ER. Se diferencia de otras propuestas en que no propone su propia notación gráfica para el diseño conceptual del DW, sino que emplea ER.

Más recientemente, cabe destacar el trabajo [5], donde se propone otro método para el diseño de DW. Este método está basado en el modelo MD IDEA y propone una serie de procesos que cubren el diseño conceptual, lógico y físico de un DW. Una de las principales ventajas con respecto a las propuestas anteriores es el hecho de que para extraer el esquema conceptual se tenga en cuenta, además de las fuentes de datos operacionales, los requisitos de usuario. Sin embargo,

este método está enfocado al modelado de datos y no abarca otros aspectos del diseño de los DW como puede ser el diseño de los procesos ETL.

Finalmente, en [4] se evalúan diversos métodos de diseño de DW y se propone un nuevo método que destaca por tener en cuenta la gestión de los metadatos. Sin embargo, carece de un modelo que permita reflejar y documentar el diseño del DW, ya que únicamente enumera una serie de actividades que se tienen que realizar para la construcción del DW.

Por tanto y, según lo comentado anteriormente, consideramos que en la actualidad no existe un método estándar, formal y riguroso ampliamente aceptado que cubra de una forma integrada todas las fases de diseño de los DW desde el modelado de datos, pasando por el diseño de los procesos ETL hasta la implementación final del DW.

## 3. Esquema general del método

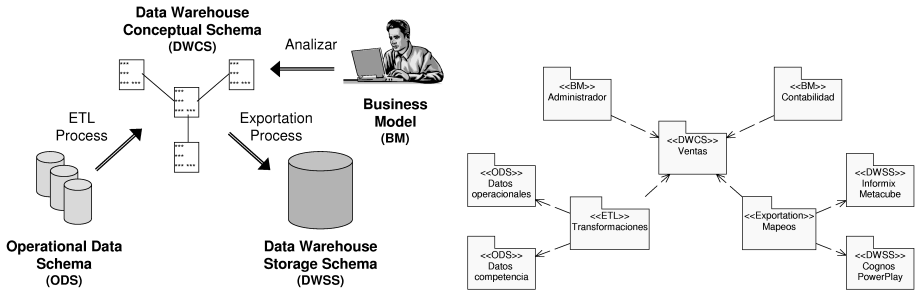
La arquitectura de un DW se suele representar como varios niveles de datos, donde los datos de un nivel se obtienen de los datos del nivel anterior [12]. Partiendo de esta arquitectura, en nuestro método consideramos que el desarrollo de un DW se puede estructurar en un modelo que integra cuatro esquemas, tal como se muestra en la Figura 1 (a):

- `Fuentes de datos operacionales (Operational Data Schema, ODS)`: define la estructura de las fuentes de datos operacionales y externas.
- `Esquema conceptual del almacén de datos (Data Warehouse Conceptual Schema, DWCS)`: define el esquema conceptual (hechos, dimensiones, jerarquías, etc.) del DW.
- `Esquema de almacenamiento del DW (Data Warehouse Storage Schema, DWSS)`: define el almacenamiento físico del DW según la plataforma destino.
- `Modelo de negocio (Business Model, BM)`: define las diferentes formas de acceder al DW desde el punto de vista del usuario final. Se compone de diferentes subconjuntos del `DWCS`.

Además, también hacen falta dos esquemas de mapeo para obtener un diseño global e integrado de un DW:

- `Procesos ETL (ETL Process)`: define el mapeo entre `ODS` y `DWCS`.
- `Procesos de exportación (Exportation Process)`: define el mapeo entre `DWCS` y `DWSS`.

Nuestro método logra integrar cada uno de los esquemas y mapeos previamente presentados. Para ello, usamos una notación de modelado basada en UML [20]. Cada uno de los esquemas y mapeos se representa mediante un paquete estereotipado. Por tanto, proporcionamos seis estereotipos: «ODS», «DWCS», «DWSS», «BM», «ETL» y «Exportation». Además, gracias al uso de los paquetes de UML, el diseño de DW grandes y complejos se simplifica porque el diseñador del DW puede establecer diferentes niveles de detalle en su diseño.




(a) (b)

Diferentes esquemas y mapeos empleados en el diseño de un almacén de datos

Nivel superior de un almacén de datos

**Figura 1.** Esquema general del método y ejemplo de modelo

Los diferentes esquemas (representados mediante paquetes de UML) de nuestro método se relacionan entre sí mediante dependencias de UML, tal como se muestra en el ejemplo de la Figura 1 (b) que representa el nivel superior de un DW. Una dependencia en UML se representa mediante una línea discontinua con una punta de flecha. En nuestro método se pueden establecer las siguientes dependencias:

- «ETL» → {«ODS», «DWCS»}.
- «Exportation» → {«DWSS», «DWCS»}.
- «BM» → {«DWCS»}.

En las siguientes secciones, se explica con detalle cada uno de los esquemas y mapeos de nuestro método.

### 3.1. Operational data schema (ODS)

En la actualidad, no existe una extensión de UML que haya sido ampliamente aceptada para el modelado de distintos tipos de fuentes de datos. Por ello, tenemos que emplear diferentes extensiones para modelar el `ODS` según la fuente de datos. Por ejemplo, si la fuente de datos es una base de datos relacional, usamos *Rational's UML Profile for Database Design* [19], pero si la fuente de datos es una base de datos objeto-relacional, empleamos [17].

### 3.2. Data warehouse conceptual schema (DWCS)

En trabajos previos [15,16], hemos propuesto un perfil de UML para el diseño conceptual de DW según el modelado MD. Nuestra propuesta permite representar las principales propiedades MD a un nivel conceptual, como son las relaciones muchos-a-muchos entre hechos y dimensiones, las dimensiones degeneradas, las jerarquías múltiples y de camino alternativo, etc. Nuestro perfil de UML se encuentra definido formalmente mediante reglas expresadas con *Object Constraint*

*Language* (OCL) [20], que definen el correcto uso de los nuevos elementos de modelado, con lo que se evita un uso arbitrario del perfil.

Además, nuestro perfil también incluye el uso de los paquetes de UML. Gracias a ello, cuando se modelan DW grandes y complejos, no estamos restringidos a diagramas de clases planos. Nuestra propuesta establece el proceso de diseño en tres niveles (la Figura 2 muestra un resumen de nuestra propuesta):

**Nivel 1** : Definición del modelo. Un paquete representa un esquema estrella de un modelo MD. En este nivel, una dependencia entre dos paquetes indica que los esquemas estrella comparten al menos una dimensión.

**Nivel 2** : Definición de un esquema estrella. Un paquete representa un hecho o una dimensión de un esquema estrella. En este nivel, una dependencia entre dos paquetes de dimensión indica que las dimensiones comparten al menos un nivel en sus correspondientes jerarquías.

**Nivel 3** : Definición de un hecho o dimensión. Se compone de un conjunto de clases que representan los niveles jerárquicos en un paquete de dimensión o el esquema estrella completo en el caso de un paquete de hecho.

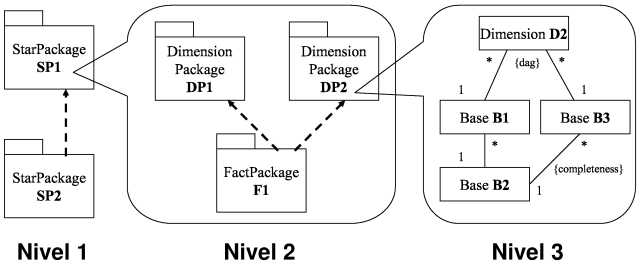



**Figura 2.** Los tres niveles de un modelo MD representados mediante paquetes de UML

En el Cuadro 1 mostramos los seis estereotipos más representativos de nuestro perfil de UML para el diseño conceptual de un DW.

### 3.3. Data warehouse storage schema (DWSS)

Según la implementación del DW que se desee realizar (relacional, MD, etc.), el `DWSS` se modela mediante la extensión de UML correspondiente (de forma similar a como se define el `ODS`).

Además, nuestra propuesta ofrece dos posibilidades de creación de este esquema: manual o automática. En la primera, el diseñador del DW necesita definir el `DWSS` y el mapeo (`Exportation Process`) entre el `DWCS` y el `DWSS`; en la segunda, un algoritmo de transformación específico para cada tipo de almacenamiento genera el correspondiente `DWSS` con el mapeo apropiado.

### 3.4. Business model (BM)

Este esquema se emplea para adaptar el DW según las distintas necesidades particulares de los usuarios finales. De este modo, se pueden crear distintas

| Concepto MD (Estereotipo) | Descripción | Icono |
|---|---|---|
| StarPackage | Paquetes de este estereotipo representan esquemas estrellas, compuestos de hechos y dimensiones; se emplea en el nivel 1 |  |
| FactPackage | Paquetes de este estereotipo representan hechos, compuestos de medidas y relacionados con las dimensions; se emplea en el nivel 2 |  |
| DimensionPackage | Paquetes de este estereotipo representan dimensiones, compuestas de jerarquías; se emplea en el nivel 2 |  |
| Fact | Clases de este estereotipo representan hechos, compuestos de medidas; se emplea en el nivel 3 |  |
| Dimension | Clases de este estereotipo representan dimensiones, compuestas de jerarquías; se emplea en el nivel 3 |   |
| Base | Clases de este estereotipo representan niveles de jerarquía en una dimensión; se emplea en el nivel 3 |   |

**Cuadro 1.** Conceptos multidimensionales y su representación en UML

"vistas" de un DW orientadas al análisis o porque no interesa que los usuarios finales tengan acceso a todo el DWCS por distintas razones (seguridad, facilidad de uso, etc.).

Para crear el BM, empleamos el mecanismo de importación de UML, que permite crear distintos submodelos del DWCS. Los distintos modelos del BM se pueden implementar como *data marts*, ya sea reales (almacenados en sus propias estructuras pobladas con datos a partir del DW) o virtuales (definidos como vistas sobre el DW).

Además, en este esquema también se definen las consultas iniciales solicitadas por los usuarios finales. Para ello, se emplean las clases cubo [23], que permiten representar de una forma visual y sencilla las consultas de usuario.

### 3.5. ETL process

En [22], hemos propuesto una extensión de UML que permite el modelado conceptual de los procesos ETL. Nuestra extensión proporciona los mecanismos necesarios para especificar las operaciones típicas de los procesos ETL, como la integración de distintas fuentes de datos, la transformación de los atributos, la generación de claves substitutas (*surrogate keys*), etc. Un proceso ETL se define combinando los distintos mecanismos que proporcionamos.

En nuestra propuesta, hemos definido un conjunto reducido pero potente de mecanismos ETL, con el fin de reducir la complejidad de nuestra propuesta y facilitar su uso.

En el Cuadro 2 mostramos un resumen de los mecanismos de nuestra propuesta, en la que los mecanismos ETL se relacionan entre sí por medio de dependencias de UML. Además, a cada mecanismo se le puede añadir una nota de

| Mecanismo ETL (Estereotipo) | Descripción | Icono |
|---|---|---|
| `Aggregation` | Agrega los datos (SUM, AVG, MAX/MIN, COUNT, etc.) en base a algún criterio |  |
| `Conversion` | Cambia los tipos de datos, el formato o calcula nuevos datos (atributos derivados) a partir de los datos existentes |  |
| `Filter` | Filtra los datos no deseados y verifica la calidad de los datos en base a restricciones |  |
| `Incorrect` | Redirige los registros incorrectos o descartados a un destino separado para su posterior verificación; sólo se puede usar con `Filter`, `Loader` y `Wrapper` |  |
| `Join` | Une dos fuentes de datos relacionadas entre sí a través de uno o varios atributos |  |
| `Loader` | Carga los datos en el destino de un proceso ETL (en un hecho o dimensión del DW) |  |
| `Log` | Controla y registra la actividad de otro mecanismo ETL, con el fin de auditar las transformaciones realizadas |  |
| `Merge` | Integra los datos provenientes de dos o más fuentes de datos con atributos compatibles |  |
| `Surrogate` | Genera una clave substituta única, que se emplea para reemplazar la clave empleada en las fuentes de datos |  |
| `Wrapper` | Transforma una fuente de datos nativa en una fuente de datos basada en registros |  |

**Cuadro 2.** Mecanismos ETL y su representación en UML

UML para explicar el funcionamiento del mecanismo y definir el mapeo entre los atributos en el origen y en el destino.

### 3.6. Exportation process

Este esquema define el mapeo entre el `DWCS` y el `DWSS`. Como se ha comentado previamente en la sección 3.3, este mapeo se puede definir de forma manual o automática mediante la aplicación de una serie de algoritmos de transformación que crean las estructuras de almacenamiento apropiadas según la plataforma de destino elegida (relacional, MD, OO, etc.).

## 4. Aplicación del método

Un buen método no se compone únicamente de una notación gráfica, sino que también debe incluir una forma de usarlo. En el Cuadro 3, presentamos un resumen de los pasos que el diseñador debería seguir para la construcción de un DW mediante nuestro método. Debido a la falta de espacio, sólo vamos a describir los principales pasos a desarrollar y no vamos a tratar temas como determinar la infraestructura necesaria para el DW, definir los participantes (roles) con sus respectivas responsabilidades, etc.

En la Figura 3 mostramos los principales pasos de nuestro método mediante un diagrama de actividades de UML. El diagrama se ha dividido en dos calles (*swimlanes*) según quién guía las actividades descritas: Usuarios finales del DW

(los usuarios finales "orientan" el trabajo de los diseñadores y administradores del DW) y Diseñadores y administradores del DW (no necesitan de la participación de los usuarios finales, ya que disponen de toda la información necesaria para realizar su labor). Las actividades donde se aplican los modelos presentados en este artículo aparecen sombreadas: con un color claro aquellas actividades donde se crean los esquemas (se indica en una esquina el esquema creado) y con un color oscuro aquellas actividades donde se crean los mapeos entre esquemas. Además, las actividades se han dividido en tres grupos según la fase de creación del DW en la que participan: análisis, diseño e implementación. Por último, las transiciones definen un orden secuencial de las actividades y también indican el empleo de información procedente de otra actividad.

- **Análisis**:
  - Determinar requisitos iniciales: se define el alcance del DW mediante entrevistas con los usuarios finales; se revisan informes ya existentes y se recopilan los requisitos iniciales de los usuarios.
  - Definir reglas de negocio: se definen diversas reglas de negocio que se aplicarán en la construcción del DW (por ejemplo, la definición de medidas derivadas como "beneficio neto" o "porcentaje de devolución de producto").
  - Identificar fuentes de datos operacionales y externas: se definen las fuentes de datos, tanto operacionales como externas (datos económicos, censos de población, etc.), que alimentarán el DW. Para ello se tienen en cuenta las necesidades expresadas por los usuarios finales. Al final de esta actividad se obtiene el `ODS`.
- **Diseño**:
  - Construir esquema conceptual: al final de esta actividad se obtiene el `DWCS`. Para acometer esta actividad existen dos estrategias "extremas", que se han representado mediante las dos transiciones con una línea discontinua: *top-down* (definir el DW según los requisitos de los usuarios finales), o *bottom-up* (definir el DW en base a los datos disponibles en las fuentes de datos).
  - Definir procesos ETL: se definen los procesos ETL como un mapeo entre las fuentes de datos (`ODS`) y el DW (`DWCS`); las reglas de negocio se aplican para calcular atributos derivados, definir transformaciones de atributos, etc. Esta actividad y la anterior definen un ciclo, ya que al crear los procesos ETL se puede detectar algún fallo en el `DWCS` (por ejemplo, que un atributo de una dimensión no exista en el `ODS`), por lo que será necesario modificar el `DWCS`.
  - Definir *data marts*: a partir de los requisitos iniciales de usuario y del `DWCS` se definen distintos `BM`, que pueden implementarse como *data marts* reales o virtuales.
  - Definir informes: las consultas iniciales se definen mediante las clases cubo.
- **Implementación**:
  - Definir almacenamiento: se define el tipo de almacenamiento empleado para el DW (relacional, MD, OO, etc.) y se crea el correspondiente esquema lógico (el `DWSS`).
  - Definir procesos exportación: se define el mapeo entre el `DWCS` y el `DWSS`. Tal como se vio previamente, este mapeo puede definirse de forma manual o automática mediante una serie de algoritmos de exportación.
  - Implementar informes: los informes solicitados por los usuarios se implementan en la herramienta de consulta empleada (generalmente una aplicación OLAP).

**Cuadro 3.** Principales pasos de aplicación del método

## 5. Conclusiones y trabajos futuros

En este artículo hemos presentado un método basado en UML que permite modelar de forma integrada las distintas partes de un DW. La principal aportación de nuestro método es proporcionar un marco global que permite modelar todos los aspectos fundamentales de los DW como son los esquemas conceptual

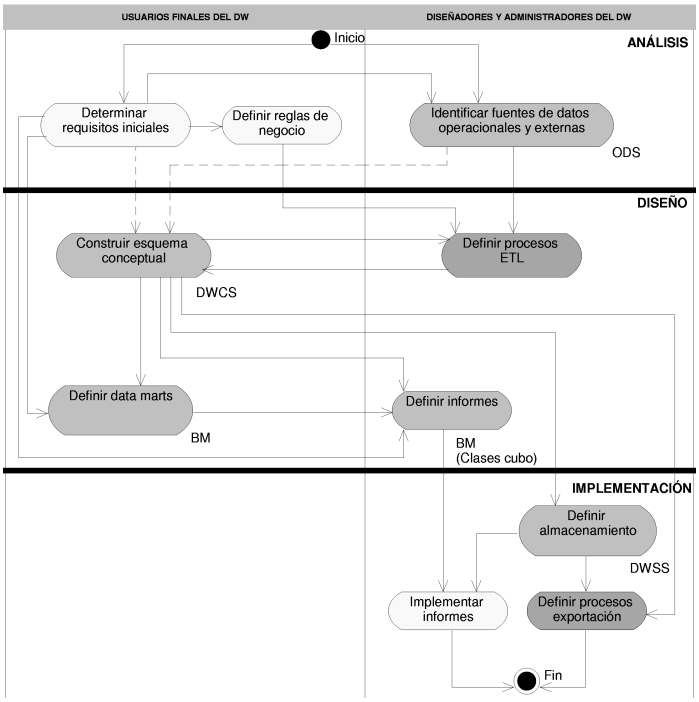



**Figura 3.** Diagrama de actividad con los principales pasos de aplicación del método

y lógico, los procesos ETL, etc. Además, gracias el empleo de los paquetes de UML, nuestro método es escalable y permite abordar el diseño de DW complejos. El aprendizaje de nuestro método se simplifica gracias al empleo de un lenguaje de modelado estándar como es UML. Por último, hemos proporcionado una serie de pasos que guían la aplicación de nuestro método.

Respecto a los trabajos futuros, estamos trabajando en la definición formal de un perfil de UML que abarque todo nuestro método y también pretendemos proporcionar una serie de guías (heurísticas) que ayuden a crear modelos correctos mediante nuestro método.

## Referencias

1. A. Abelló, J. Samos, y F. Saltor. A Framework for the Classification and Description of Multidimensional Data Models. En *Proc. of the 12th Intl. Conf. on Database and Expert Systems Applications*, páginas 668–677, Munich, Alemania, 2001.
2. A. Abelló, J. Samos, y F. Saltor. YAM2 (Yet Another Multidimensional Model): An Extension of UML. En *Intl. Database Engineering & Applications Symposium*, páginas 172–181, Edmonton, Canada, 2002.
3. L. Cabibbo y R. Torlone. A Logical Approach to Multidimensional Databases. En *Proc. of the 6th Intl. Conf. on Extending Database Technology*, vol. 1377 de *LNCS*, páginas 183–197, Valencia, España, 1998.

4. L. Carneiro y A. Brayner. X-META: A Methodology for Data Warehouse Design with Metadata Management. En *Proc. of the 4th Intl. Workshop on Design and Management of Data Warehouses*, páginas 13–22, Toronto, Canada, 2002.
5. J.M. Cavero, M. Piattini, y E. Marcos. MIDEA: A Multidimensional Data Warehouse Methodology. En *Proc. of the 3rd Intl. Conf. on Enterprise Information Systems*, páginas 138–144, Setubal, Portugal, 2001.
6. M. Corey, M. Abbey, I. Abramson, y B. Taub. *Oracle8i Data Warehousing*. Oracle Press. Osborne/McGraw-Hill, 2001.
7. N.T. Debevoise. *The Data Warehouse Method*. Prentice-Hall, New Jersey, USA, 1999.
8. S.R. Gardner. Building the Data Warehouse. *Com. of the ACM*, 41(9):52–60, 1998.
9. M. Golfarelli y S. Rizzi. A methodological Framework for Data Warehouse Design. En *Proc. of the ACM 1st Intl. Workshop on Data Warehousing and OLAP*, páginas 3–9, Bethesda, USA, 1998.
10. B. Husemann, J. Lechtenborger, y G. Vossen. Conceptual Data Warehouse Design. En *Proc. of the 2nd Intl. Workshop on Design and Management of Data Warehouses*, páginas 3–9, Estocolmo, Suecia, 2000.
11. W.H. Inmon. *Building the Data Warehouse*. QED Press/John Wiley, 1992.
12. M. Jarke, M. Lenzerini, Y. Vassiliou, y P. Vassiliadis. *Fundamentals of Data Warehouses*. Springer-Verlag, 2 edición, 2003.
13. R. Kimball. *The Data Warehouse Toolkit*. John Wiley & Sons, 1996.
14. R. Kimball, L. Reeves, M. Ross, y W. Thornthwaite. *The data warehouse lifecycle toolkit*. John Wiley, 1 edición, 1998.
15. S. Luján-Mora, J. Trujillo, y I. Song. Extending UML for Multidimensional Modeling. En *Proc. of the 5th Intl. Conf. on the Unified Modeling Language*, vol. 2460 de *LNCS*, páginas 290–304, Dresden, Alemania, 2002.
16. S. Luján-Mora, J. Trujillo, y I. Song. Multidimensional Modeling with UML Package Diagrams. En *Proc. of the 21st Intl. Conf. on Conceptual Modeling*, vol. 2503 de *LNCS*, páginas 199–213, Tampere, Finlandia, 2002.
17. E. Marcos, B. Vela, y J.M. Cavero. Extending UML for Object-Relational Database Design. En *Proc. of the 4th Intl. Conf. on the Unified Modeling Language*, vol. 2185 de *LNCS*, páginas 225–239, Toronto, Canada, 2001.
18. D. Moody y M. Kortink. From Enterprise Models to Dimensional Models: A Methodology for Data Warehouse and Data Mart Design. En *Proc. of the 3rd Intl. Workshop on Design and Management of Data Warehouses*, páginas 1–10, 2001.
19. E.J. Naiburg y R.A. Maksimchuk. *UML for Database Design*. Object Technology Series. Addison-Wesley, 2001.
20. Object Management Group (OMG). Unified Modeling Language Specification 1.4. Internet: http://www.omg.org/cgi-bin/doc?formal/01-09-67, 2001.
21. C. Sapia, M. Blaschka, G. Höfling, y B. Dinter. Extending the E/R Model for the Multidimensional Paradigm. En *Proc. of the 1st Intl. Workshop on Data Warehouse and Data Mining*, vol. 1552 de *LNCS*, páginas 105–116, Singapore, 1998.
22. J. Trujillo y S. Luján-Mora. A UML Based Approach for Modeling ETL Processes in Data Warehouses. En *Proc. of the 22nd Intl. Conf. on Conceptual Modeling*, vol. 2813 de *LNCS*, Chicago, USA, 2003.
23. J. Trujillo, M. Palomar, J. Gómez, y I. Song. Designing Data Warehouses with OO Conceptual Models. *IEEE Computer*, 34(12):66–75, 2001.
24. N. Tryfona, F. Busborg, y J.G. Christiansen. starER: A Conceptual Model for Data Warehouse Design. En *Proc. of the ACM 2nd Intl. Workshop on Data Warehousing and OLAP*, Kansas City, USA, 1999.