ISSN 1409-2441

Ingeniería

Revista de la Universidad de Costa Rica

Enero/Diciembre 1998 VOLUMEN 8 Nos. 1 y 2

Ingeniería 8(1,2): 141-153, 1998 San José. Co;ta Rica

EL USOn,EL,MODELO ENTIDA.D-RELACION EN EL DISEÑO DE LA REPRESENTACIÓN DE DATOS ENLAS:B~SES DE DATOS ORIENTÁDASA OBJETOS

Elzbieta Malinowski G.[1]

Resumen

El modelo entidad-relat:ión es el modelo conceptual más usado para el diseño de bases de datos relacionales. Sin embargo, el desarrollo de los manejadores de las bases de datos orientadas a objetos aunque abrió nuevas posibilidades para los sistemas de bases de datos, no resuhó una ahemativa de fácil transición como para que los diseIiadores los usen. El presente artículo describe las caracterí;ticas generales de las bases de datos orieIltadas a objetos. Además, se mencionan algunos métodos desarrollados para su análisis y diseño. También se ofrece la posibilidad de utilizar el modelo eIltidad-relación y se e;pecifican, de una forma similar al modelo relacional, los pasos necesarios para pasarlo a la representación de clases (solamente la parte de datos) en las bases de datos orientadas a objetos. Como ejemplo de aplicación de estos pasos se presenta el modelo de entidad-relación y la defmición de datos en forma de clases desarrolladas para la base de datos utilizada en colecciones biológicas.

Summary

The entity-relationship model is tbe one most used in tbe relational database desigp.. However, even tbough tbe development of tbe object-oriented database management sy;tems opens /lew possibilities to tbe database SysteIIlS, ,till it doe;n't olfer "soft" transition for database desigp.ers to switch to tbis new paradigm. This article describes general characterí;tics of tbe object-oriented databases. Moreover, it mentions some of tbe methods developed for object-oriented analisys and designo Also, it olfers !he possibility to use tbe entity-relationshiop model and it specifies necessary steps (similar to specifications of steps for relational database) to transform tbis model to tbe class implementation (only tbe data part) for object-oriented databases. As an example of implementation of tbese steps tbis article presents tbe entity-relationship model and class defmition developed for database ofbiological collections.

1. INTRODUCCIÓN

La presentación del modelo relacional hecha por Codd en el año 1970 [Codd, (1994)] causó un impacto enorme en los conceptos establecidos relacionados con las bases de datos (BD). El enfoque relacional, como cualquier enfoque lluevo, encontró sus defensores y atacantes ya que la mayoría de los sistemas de manejo de bases de datos (DBMS, *Database Management System)* usaban en esa época el enfoque de redes o jerárquico.

Sin embargo, como este modelo es fuertemente dependiente del nivel de la implementación, Chen en el año 1980 [Chen, (1994)] presentó un modelo abstracto llamado modelo entidad-relación (ER) que facilita el diseño del modelo relacional sin preocuparse de los detalles de implementación. Este modelo fue posteriormente ampliado por Teorey [Teorey et al., (1986)] y contiene notaciones que diferencian las jerarquías de especialización y generalización. También Elmasri y Navathe en 1989 propusieron algunas extensiones a este modelo Elmasri, (1994). En general, los autores en sus extensiones se acercaron más y más hacia la presentación de tipos abstractos, de una forma muy parecida a la orientación a objetos.

Siguiendo los cambios paradigmáticos de los lenguajes de programación, en los años 90s otra tendencia de modelaje por objetos comenzó a ser más usada en el diseño de sistemas siendo una visión más natural del sistema real [Yourdon, (1990)]. Los lenguajes

[1] Profesor, M.Sc. Ese. de Ciencias de la Computación e Informática, UCR

de programación por objetos incorporaron, entre otros, los conceptos de encapsulación, polimorfismo, herencia, y permitieron modular los elementos, dejando los detalles de implementación invisibles para el usuario; también ha disminuido la cantidad de código necesaria para escribir lila aplicación y en consecuencia, ha disminuido el tiempo de depuración de programas, ofreciendo además la posibilidad de reutilizar el código.

Aunque los lenguajes de programación actualmente adoptaron el paradigma de orientación por objetos, las bases de datos encontraron dificultades con respecto a la persistencia de los objetos. Atkinson [Atkinson, (1994)] mencionó que se debe contestar rápidamente a quién corresponde el problema de persistencia de datos: a los investigadores de lenguajes de programación o a los investigadores de BD. Respondiendo a este reto, se han desarrollado varios sistemas de manejo de bases de datos orientados a objetos (OODBMS, *Object-Oriented Database A1anagement Systems)* que aseguran la persistencia automática de datos, como *ObjeclStore, Shore, O_2, Orion, GemSlone* [Kim, (1994)], [Lamb, (1994)], [Deux, (1994)], [Bertino, 1995]. AsÍ, se ha tomado consciencia que las bases de datos orientadas a objetos (BDOO) no son una cuestión temporal, sino el camino al progreso en BD.

Sin embargo, para poder utilizar un sistema de BDOO es necesario hacer un adecuado análisis y diseño. De acuerdo con Sommerville [Sommerville, (1992)] un diseño conceptual eficiente es la base principal para asegurarse que los detalles del funcionamiento del sistema son entendidos y captados en la primera parte del diseño. donde los costos de las correcciones no son todavía significativos; de esta forma se puede disminuir el costo de mantenimiento que es actualmente el mayor rubro de porcentaje en el desarrollo del sistema.

Los métodos y técnicas para diseño convencional (estructurado), especialmente el modelo ER, son bien conocidos. En contraste, aunque existe un número considerable de métodos de análisis y diseño por objetos, ninguno de ellos es tan conocido y usado como lo es el modelo ER.

Según Graham [Graham, (1996)] los métodos de análisis y diseño orientado a objetos se pueden clasificar en dos categorías: los que reproducen métodos estmcturados ya existentes, utilizando tres notaciones distintas: para datos, dinámicas y procesos, y los otros que afirman que los objetos combinan los procesos y los datos y por esta razón es necesaria solamente una notación.

En este artículo, tomando como ejemplo la BD de colecciones biológicas de anfibios y reptiles realizada en el proyecto de investigación titulado: "Diseño e implementación de la base de datos de las colecciones biológicas para las consultas interactivas en la red "World Wide Web" " se presentará la posibilidad de representar los datos en OODBMS basándose en el modelo conceptual el modelo ER.

En las siguientes secciones se presentan las características generales de BDOO y se mencionan algunos métodos de análisis y diseño orientado a objetos. Además, se describen conceptos generales presentados en el modelo ER, proponiendo los pasos necesarios para pasarlo en forma automática a la definición de los atributos de las clases usadas en el enfoque de la orientación a objetos. Finalmente se presenta como ejemplo un modelo ER y se definen las clases con sus respectivos atributos.

2. CARACTERÍSTICAS BÁSICAS DE BASES DE DATOS ORIENTADAS A OBJETOS

Es muy dificil establecer las caracteristicas generales de BDOO. Una de las razones es que aunque Atkinson [Atk:inson, (1992)] propuso algunas reglas para unificar los conceptos relacionados con BDOO, todavía no existe 1m acuerdo entre los investigadores inclusive hasta en la terminología a usar. Resumiendo la información de diferentes origenes, podemos

definir en general las siguientes características que posee una BDoo:

- Ozsu y Valduriez [Ozsu, (1991)] dicen que el objeto es un par de valores (id. v). donde "id" es el identificador creado por el sistema y v indica el valor que objeto puede tener. Otros autores ([Elmasri, (199-1-)], [Masunga. (1991)]) definen el objeto como un triple de valores (id, tipo, v). donde el tipo representa una de las estructuras definidas por el sistema (átomo, conjunto. tupla. y otros). Algunos autores usan el término "estado" en lugar de "valor" [Kim.(l991)], [Khoshafian, (1993)].
- Cada objeto tiene un identificador definido por el sistema.
- Cada objeto tiene su memoria privada y conjunto de operaciones (métodos). La memoria privada tiene el valor de los atributos [Masunga. (1991) ▌ Khoshofian [Khoshofian, (1993)] nombra estos atributos como las variables de instancia. Existen dos acercamientos con respecto a la existencia de objetos y valores en BDoo:
  - en el acercamiento usado en Smalltalk [Golberg. (1983)] cada valor es representado como objeto que tiene su propio identificador y puede ser accesado solo por medio del método.
  - en el otro acercamiento se pennite tener valores y objetos [Elmasri. (1994)]. [Lecuse. (1992)], [Khoshafian. (1993)]. [Kanellakis. (1992)].
- La encapsulación permite la posibilidad de definir las operaciones y funciones para manipular el objeto o devolver su estado.
- El uso de los constructores permite crear objetos de una complejidad arbitraria. Los constructores básicos son el átomo. tupla y conjunto. También, la lista, arreglo y "bag" son los otros tipos usados comúnmente [Elmasri. (1994)]. Existen así llamados objetos complejos donde el valor del objeto es representado por el identificador de otro objeto [Ozsu, (1991)].
- El valor del objeto depende del tipo:
  - si el objeto es atómico su valor es definido por el dominio especificado por el usuario o por el sistema de DBMS.
  - si el objeto es el conjunto, su valor es representado por el conjunto de diferentes identificadores de los objetos.
  - si el objeto es la tupla, el valor está formado por (at_1 :id!, at_2:id_2, ... ,at_n:idn). donde at_i indica nombre del atributo, y id_i indica un identificador del objeto.
- Los objetos se agrupan en clases.
- Las clases se puede agmpar en superclases *(bottom-up)* o dividir en subclases *(top-down)* usando así la estmctura jerárquica y herencia, donde se pueden heredar los atributos y las operaciones.

Los sistemas orientados a oQjetos crean un importante cambio permitiendo a los usuarios definir sus propios tipos o métodos de acceso. También aceptan la evolución del esquema, *lime lrmJeling* y además se desempeñan bien en las aplicaciones no tradicionales donde DBMS relacional baja su rendimiento. Sin embargo. estos sistemas tienen varios problemas pendientes de resolver como son la igualdad de los oQjetos [Masunga, (1991)]. las consultas (OSQL todavía no tiene una forma completa y fiable) rAnnevelik. (1995)], operaciones "join" [Tanaka, (1990)], modificaciones del esquema [Kim. (1990»), control de concurrencia [Martin, (1994)], vistas rSholl, (1994)] Yotros.

Para solucionar temporalmente algunos de estos problemas se elaboraron cambios en los sistemas relacionales que permitieron a usuario crear la imagen de trabajar con los sistemas orientados a objetos y así mantener las ventajas de bases de datos relacionales. Estos DBMS se llaman extensibles. híbridos u objeto/relacionales [Bontempo, (1995)].

Las BD extensibles (yen consecuencia orientadas a oQjetos) son un paso natural en el desarrollo de DBMS, porque permiten muchas ventajas con respecto a las bases de datos relacionales. Cabe mencionar algunas de ellas:

- No redundancia en los datos.
- Facilidad de presentar relaciones complejas en forma de ligas entre objetos.
- Mejor correspondencia entre el mundo real y el diseño.
- Posibilidades de usar los tipos y métodos de acceso definidos por el usuario.
- Reutilización del código.
- Existencia de un identificador Único para cada objeto que permite no usar llaves primarias obligatorias - y a veces forzadas - como en los sistemas relacionales.

Las BDOO son una nueva tendencia en ocasiones dificil de asimilar por el personal encargado de BD. Una de las razones es la dificultad relacionada con el aprendizaje de nuevos métodos de diseño orientado a objetos.

3. TRABAJOS RELACIONADOS CON LOS MÉTODOS DEL ANÁLISIS Y DISEÑO ORIENTADO A OBJETOS y LAS CARACTERISTICAS DEL MODELO ENTIDAD-RE LACIO N

Haciendo una revisión bibliográfica se puede encontrar diferentes métodos de análisis y diseílo orientado a objetos. Se está generalmente de acuerdo que estos métodos son incompletos [Graham, 1996], sin embargo, esto no afecta que algunos de ellos sean bien conocidos entre los investigadores.

El método de CoadIYourdon [Coad, (1991)] propone hacer análisis en cinco fases llamadas temas, objetos estructuras, atributos y servicios y ampliarlas con cuatro componentes más en el proceso de diseño para obtener el así llamado "Componente del Dominio de Problema".
Otro acercamiento que propone Booch [Booch, 1996]) ofrece tratar al principio los aspectos estáticos para describir la estructura lógica y la estructura fisica usando diagramas de clases, de objetos, de módulos y de procesos. Posteriormente se diseña la dinámica de clase y de instancias por medio de diagramas de transacción de estados y diagramas temporales respectivamente.

La Técnica de Modelado de Objetos (OMI, *Object Modelling Technique)* [Rumbaugh, (1991)] es considerada como uno de los sistemas más completos que se han publicado hasta este momento y tiene sus raíces en los métodos estmcturados tradicionales. OMI consta de tres fases: análisis, diseño de sistemas y diseño de objetos, que incluye la construcción de diagramas similares a los de CoadIY ourdon. Aquí, la notación está principalmente basada en el modelo ER con las operaciones que se han añadido a los iconos de entidades. Después se construye un modelo dinámico y de diagrama de flujo globales de sucesos. El tercer paso es el modelo funcional.

También existe una variedad de otros métodos como: MartinJOdell-Ptech, Desfray, Os:4 *(Objecl Sistem Analysis), Texel, CRC* y otros [Graham, (1996)].

Sin embargo, actualmente la situación real encontrada es que la mayoria de los modelos conceptuales, si existen, están desarrollados usando el modelo ER y aunque se entienden las ventajas de usar las BDOO sería muy costoso aprender un nuevo método de análisis y diseño y rediseñar el sistema de las BD de acuerdo a este. Por su simplicidad de entendimiento y aplicación, el modelo ER y sus variaciones son comúnmente usados para el diseño conceptual de las BD. Además, varias herramientas automatizadas tienen incorporados sus conceptos [Elmasri, (1994)]. Siendo uno de los modelos más utilizados entre los diseñadores de BD relacionales, el modelo ER podría ser usado como un modelo de alta abstracción para las BDOO y entonces muchos de los sistemas diseñados e implementados en DBMS relacionales no tendrían que ser rediseñados en el caso de pasarlas a BDOO. Además, en el caso cuando no existe el modelo conceptual, y existe solo el modelo de implementación (relacional), se pueden encontrar herramientas automáticas o se puede aplicar en forma manual la

"ingenieria inversa" para obtener el modelo ER.

Este artículo propone usar el modelo ER para crear clases (parte de los atributos, datos) en BDOO y de esta forma aplicar una "transición suave de incorporar la orientación a objetos a métodos convencionales" [Graham, (1996), pg.249].

El modelo ER no es un modelo estándar. Por esta razón se pueden encontrar en la literatura diferentes representaciones gráficas que se utilizan para el diseño. En generaL se distinguen los siguientes elementos que pueden existir en un modelo ER:

- Entidad: objeto que puede diferenciarse de otros objetos del mundo real.

- Tipo de entidades: conjunto de entidades, que tienen los mismos atributos. Existen dos clases de tipo de entidad:

Débil: tipo de entidad que no tiene atributo llave propio.
Fuerte: tipo de entidad que posee un atributo llave propio

- Relación: conjunto de asociaciones entre varios tipos de entidades.

- Atributo: propiedad particular que describe una entidad o relación. Tipos:
  - Simple: no es divisible.
  - Compuesto : puede ser dividido en partes
    más pequeñas.
  - Valor único: tiene un solo valor para una
    entidad específica.
  - Multivalor : puede tener un conjunto de
    valores para la misma entidad.
  - Derivado : puede ser determinado o derivado
    de otro atributo o de entidades relacionales.
  - Llave : atributo cuyos valores son distintos
    para cada entidad.

- Restricciones estmcturales:
  - cardinalidades de mapeo : el número de entidades con las que puede asociarse otra entidad mediante relación (una a una, una a muchas, muchas a una y muchas a muchas)
  - participaciones - indicación. si todas las entidades participan en la relación por lo menos una vez (parcial y total)

- Generalización: unión de dos o más tipos de entidades (de bajo nivel) para producir un tipo de entidad de alto nivel. Los atributos de alto nivel se heredan a las entidades de bajo nivel.

- Especialización: formación de un tipo de entidades de bajo nivel a partir de las entidades de alto nivel. Los atributos de alto nivel se heredan a las entidades de bajo nivel.

Si se toma en cuenta solo la parte de datos del enfoque orientado a objetos, se puede ver algunas similitudes entre el modelo ER y la BDOO , dando así la posibilidad de especificar los pasos que penniten crear las clases a partir del modelo ER.

4. PASOS GENERALES PARA PASAR DE EL MODELO ENTIDAD-RELACIÓN AL MODELO ORIENTADO A OBJETOS

El modelo ER (también algunas de sus extensiones) está adaptado para usarlo en fornla eficiente en la implementación relacional y tiene desarrollado los algoritmos para pasarlo en forma automática al modelo relacional [Elmasri, (1994)], [Korth, (1994)].

Como se mencionó en la sección anterior, se puede ver que existen similitudes entre el modelo ER y el modelo de BDOO que podrían simplificar el paso del modelo ER al modelo de por objetos.

Esta relación se puede observar muy claramente con respecto a las entidades:

- Una entidad fuerte puede ser vista como un objeto.
- Una entidad débil puede tener como uno de los atributos el identificador del objeto "fuerte" del cual el objeto "débil" depende o el objeto "fuerte" puede contener identificadores de sus objetos "débiles".
- El conjunto de entidades puede representarse por medio de las clases..
- Los atributos no multivalor que describen las entidades pueden ser atributos de cada objeto.
- Los atributos multivalor se presentan como un atributo tipo conjunto dentro del mismo objeto.
- Los atributos compuestos se presentan como un atributo tipo tupla dentro del mismo objeto.

Siguiendo estás reglas cada conjunto de entidades con sus respectivos atributos formaria la parte estructural (sin tomar en cuenta el aspecto de procesos y dinámica) de una clase en BDOO.

Con respecto a las relaciones se deben seguir las siguientes reglas:

- Las relaciones 1:1 se describen como un atributo complejo, esto es si tenemos dos entidades S y T que tienen la relación 1:1 entre ellas, se crea en la entidad S (suponiendo como en el modelo relacional que esta entidad tiene participación total) un atributo cuyo valor es identificador del correspondiente objeto de la entidad T. Para facilitar la navegación entre objetos, en el objeto que corresponde a la entidad T se puede crear un atributo que contiene la referencia al correspondiente objeto de S, usando la así llamada referencia inversa, por ejemplo desarrollada en el OODBMS *ObjeclStore* [Lamb, (1994)]. Esto asegura el chequeo de consistencia de los datos en forma automática (un concepto parecido a la integridad referencial en BD relacionales). Sin embargo, esto es bastante costoso [Rumbaugh, (1991)]. Además, si la relación tiene sus propios atributos. éstos formarían parte de la tupla junto con el identificador del objeto S.
- Las relaciones l:N entre entidades S y T se reflejan creando en el objeto del lado N (el oQjeto T) un atributo complejo que es el identificador del respectivo objeto S. Si la cantidad de objetos de lado T es grande. se recomienda crear del lado S un atributo tipo conjunto que contiene los identificadores de los correspondientes objetos T. Al igual que antes se puede usar (si está disponible) el concepto de referencia inversa. Si la relación contiene atributos. en lugar de crear el objeto complejo del lado T se creará la tupla que contiene el identificador del objeto S arriba mencionado y además los atributos de la relación.
- Las relaciones N:M entre entidades S y T se presentan creando en cada uno de los oQjetos un atributo tipo conjunto. Si la relación contiene atributos. en uno de los objetos (el cual se consultará más con respecto al atributo de la relación) en el lugar de atributo tipo conjunto se crea un objeto que es conjunto de tuplas. Cada tupla contiene el identificador del objeto de otra entidad y los atributos de la relación. Otra opción para este tipo de relaciones podría ser crear un objeto que contiene conjunto de tuplas, donde cada tupla contiene identificador del objeto S. identificador del oQjetoT y los atributos de la relación. También aquí *se* puede aplicar el concepto de referencia inversa.

- Las relaciones de grado n>2 se reflejan creando un objeto que contiene los identificadores de los objetos que participan en la relación. Si la relación contiene los atributos, éstos se incluyen en dicho objeto. Se debe crear un oQjeto que contiene los elementos mencionados anteriormente para cada triple de objetos participantes en la relación

Los conceptos de especializaciones ygeneralizaciones se aplican en forma directa como las clases y las subclases.

Como se mencionó al prinCÍpIO, el enfoque presentado apoya los métodos estructurados ya existentes, utilizando tres notaciones distintas para datos, dinámicas y procesos. La parte presentada en este artículo podría servir para presentar los datos en el enfoque de BDOO. Es necesario ampliarla con el análisis y el diseño de dinámicas y procesos, para incluir los métodos necesarios por medio de los cuales se deben de comunicar los objetos.

Aunque no se cuenta con un sistema comercial de BDOO, se aplicaron los pasos anteriores al modelo ER diseñado para BD de colecciones biológicas.

5. EJEMPLO DE BASES DE DATOS DE ANFIBIOS Y REPTILES Y LA PRESENTACIÓN DE DATOS **POR** MEDIO DE LOS OBJETOS

De acuerdo a la información recopilada de los usuarios se diseñó el modelo ER para la base de datos de anfibios y reptiles presentado en la Figura N° 1.

Aplicando los pasos descritos por Elmasri y Navathe [Elmasri, (1994)] se puede pasar en forma automática este modelo al modelo relacional o si fuese necesario a modelo de redes o jerárquico, aplicando los pasos presentados por Korth [Korth, (1995)].

Utilizando el algoritmo para pasar el modelo ER a BDOO se obtienen las clases con sus atributos (se usa un lenguaje orientado a o~ietos genérico). Esta definición de clases puede ser mejorada y refinada si se dispone de una OODBMS específica. Aquí, se supone la disponibilidad de referencia inversa (como en el sistema de *ObjectStore).* Sin embargo, aunque de esta forma se puede asegurar el control automático de la consistencia de datos, puede aumentar el tiempo de respuesta a las consultas.

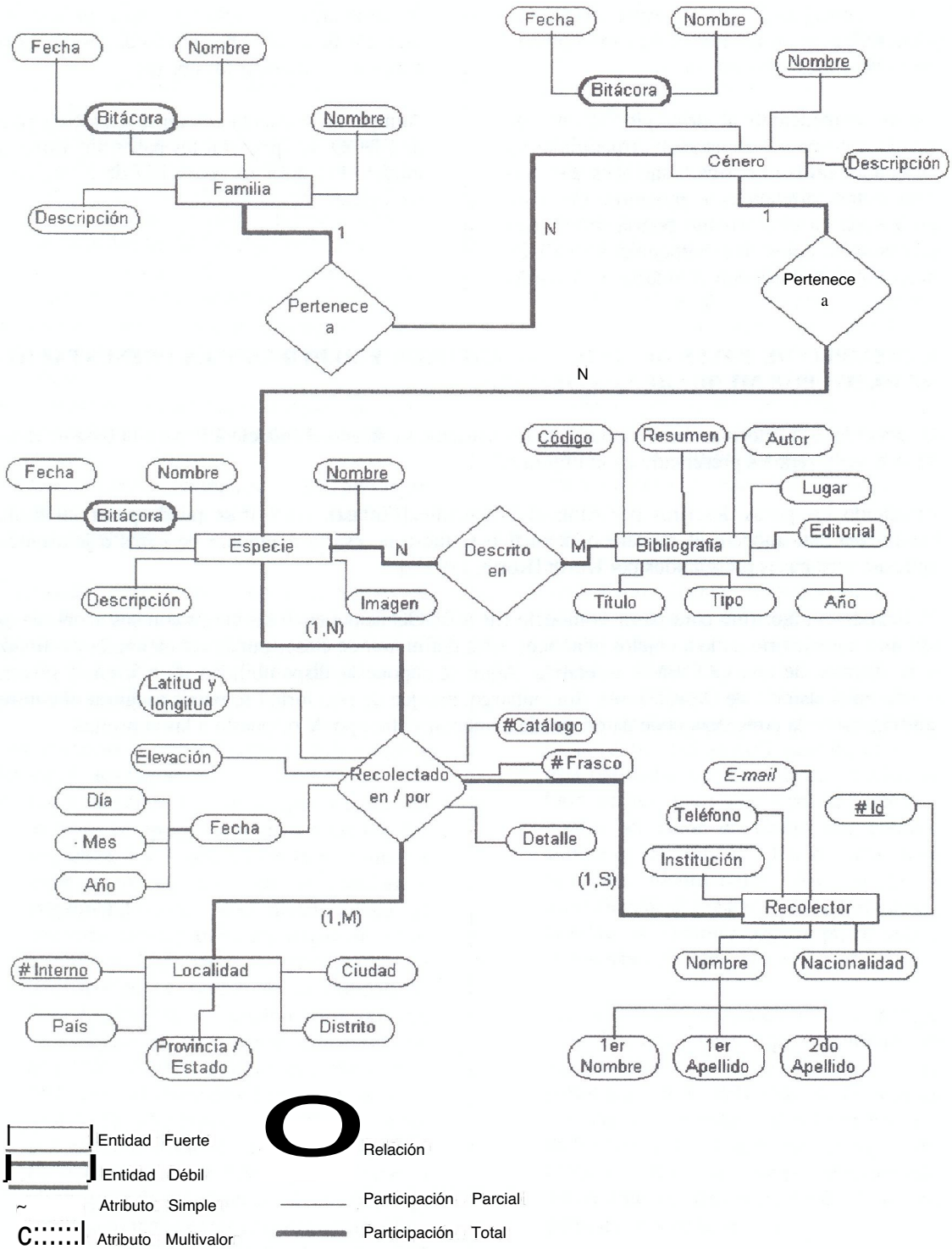

Figura N° 1. Modelo ER

```
Class Familia {

        Nombre _Actual:          slring:
        Descripción:                    slring:
        Bitácora:                       sel o[ tuple (fecha: date, nombre: string);
        Miembros _Genero *:             se! o[Genero inverse _member

Genero:: Nombre_Actual;

};

Class Genero {

        Nombre _Actual:          slring;
        Descripción:                    string:
        Bitácora:                       se! oftuple (fecha: date, nombre: slring) ;
        Pertenece _Fam:          Familia inverse _member Familia:Nombre _Actual;
        Miembros _Especie*:             set ~[Especie inverse Jnember

Especie: :Nombre_actual:

};

Class Especie {

        Nombre _Actual:          slring;
        Descripción:                    string:
        Imagen:                  string:
        Bitácora:                       ser oftuple (fecha: date. nombre: string);
        Pertenece _Genero:              Genero inl'erse Jnember Genero: :Nombre_ActuaL
        Descrito:                       set ofBibliografia inverse _member

Bibliografia: :Cod_ Ref:

};

Class Recolector {

        #ID:                            integer:
        Prim _Apellido:          string:
        Seg_ Apellido:                  string:
        Nombre:                         sfring:
        Nacionalidad:                   srring:
        E-mail:                  srring;
        Teléfono:                       phone;
        Institución:                    srring;

};

Class Localidad {
```

```
        #Localidad:                 inleger;
        País:                       string;
        Provincia:                  string;
        Ciudad:          slring;
        Distrito:        string;

};
```

Declaración de clases

Class Bibliografia {

```
        Cad _Ref:                   slring;
        Autor:                      slring;
        Titulo:                     slring;
        Ano:                        dale;
        Editorial:                  string;
        Lugar:                      string;
        Resumen:                    string;
        Describe:                   set o(Especie  inverse _member

Especie: :Nombre_ Actual:

};
```

Class Recolectado {

```
        Que:                        Especie inverse _member Especie: :Nombre_ Actual;
        Quien:                      Recolector inverse _member Recolector: :#ID;
        Donde:                      Localidad inverse _member Localidad:: #Localidad:
        #Catalogo:                  inleger;
        #Frasco:         inleger;
        Detalle:         string;
        Latitud:         decimal (5,2);
        Elevación:                  decimal (5.2);
        Fecha:                      date;
        Tipo_Bosque:                string;

};
```

* solamente si las consultas sobre este aspecto son frecuentes.

Declaración de clases (continuación)

Como se puede ver se logró independizar la fase de diseti.o conceptual de la fase de la implementación. Dependiendo de qué tipo de sistema DBMS se desea usar se pueden crear tablas o clases de acuerdo al enfoque relacional u orientado a objetos respectivamente.

Es obvio que esta fase debe ser seguida por el análisis dinámico y de procesos para establecer

los métodos de comunicación entre objetos y el manejo de excepciones.

6. CONCLUSIONES

Las BD relacionales y DBMS que permiten el manejo de los datos han servido por mucho tiempo como una herramienta eficiente de procesamiento automático de información Aunque las BD de poca complejidad se pueden desarrollar en forma inmediata usando DBMS especifica, se presentaron los modelos conceptuales para asegurar el funcionamiento adecuado y la posibilidad del futuro crecimiento del sistema. como por ejemplo el modelo entidad-relación que permite captar los aspectos importantes de un sistema de BD sin preocuparse de los detalles de la implementación. Además, siendo las BD relacionales tan usadas se diseñaron y automatizaron los algoritmos para pasar el modelo ER a tablas relacionales. Con la aparición de BDOO entramos en la confusión de poder usar el modelo ER en el diseño de la parte estática del sistema (datos). Como todo cambio de paradigma, también el paradigma de orientación a objetos hizo que se diseñaran nuevos métodos de análisis y diseño orientado a objetos, algunos de los cuales aprovechando el conocimiento existente en el análisis y diseño estructurado, otros proponiendo enfoques totalmente nuevos. Estos métodos todavía no se utilizan con la suficiente apertura como para recomendar a uno de ellos como el mejor. El general, lo que tienen estos métodos en común, es la dificil asimilación por parte de los diseñadores, especialmente los que tienen experiencia en BD relacionales y el modelo ER.

Para hacer posible la transición inevitable de BD relacionales a orientadas a objetos se propuso en este artÍCulo continuar el uso del modelo ER y aplicar los pasos que permiten en forma automática crear las clases necesarias para capturar los datos que deben ser incorporados en la BDOO. El enfoque propuesto no es definitivo. Los conceptos como referencias inversas, aunque permiten más flexibilidad, en el momento de usarlos en el sistema OODBMS pueden hacer su funcionamiento más lento de lo esperado.

El enfoque propuesto en este artículo cubre solo una parte de diseño orientado a objetos: los datos. Es necesario seguir con un modelo dinámico y funcional. De acuerdo a Rumbaugh "el modelo funcional especifica lo que sucede, el modelo dinámico cuándo sucede y el modelo de objetos especifica a qué le sucede" [Rumbaugh. (1991), pg. 171]. Limitando el diseño solo al modelo de objetos el sistema sería incompleto. El diseño funcional y dinámico nos permitirá diseñar métodos y aplicaciones necesarias para la comunicación de los objetos y el funcionamiento correcto del sistema.

7. BIBLIOGRAFIA

I. Annevelik 1. et al. *Objecl SQL -.4 Language far [he Design and fmp/emenlalion o(Object Databases* en W. Kim Modem Database Systems. Addison-Wesley, 1995.

2. Atkinson M.et al. *The Object-Oriemed Database AJanifeslo* en F.Bancilhon, C. Delobel y P. Kanelakis (eds.) Building and Object Oriented Database System. The Story of O_2. Margan Kaufman Publishers. 1992

3. Atkinson M. *Persistent FundationsfÓr Scalable Alu/ti-Paradigma S:vstems* en M.T. Ozsu, U. Dayal y P.Valduriez (edsJ Distributed Object Management. Morgan Kaufman Publishers, 1994.

4. Bertino E. y Martino L. *Sistema". de Bases deDatos Orientadas a Ohjetos.* Addison-Wesley, 1995.

5. Bontempo Ch. y Saracca C.M. *Database lvfanagement: PrincipIes ami Products.* Prentice-Hall PTR. 1995.

6. Booch G. *Análisis y Diseño Orientado a Objetos con Aplicaciones.* Addison-Wesley, 1996.

7. Chen P.P. *The Entity-Relationship Jvfodel Toward a Unified View of Data* en M. Stonebraker Readings in Database Systems. Morgan Kaufman Publishers, 1994.

8. Coad P. y Yourdon E. *ObjeclOriented Anali.5ys*. Yourdon Press/Prentice Hall, 1991

9. Codd E.F. *A relational Model of Data jar Large Shared Bankes* en M. Stonebraker Readings in Database Systems. Margan Kaufman Publishers, 1994.

10. Deuxo. *The Slory cf02* en M.Stonebraker Readings in Database Systems. Morgan Kalúman Publishers, 1994.

11. Elmasri R. y Navathe S.B. *Fundamenlals of Database S:vstems*. The BenjaminJ Cummings Publishing Company, 1994.

12. Golberg A. y Robson D. *Smalllalk -80. The Language and ils Implementation*. Addison-Wesley, 1983.

13. Graham, 1. *Métodos y modelos orienlados a objetos*. Addison-Wesley, 1996.

14. Kanellakis P. et al.. *Introduction lo Dala Model* en F.Bancilhon, C. Delobel y P. Kanelakis (eds.) Building an Object Oriented Database System. The Story of O_2. Margan Kaufman Publishers, 1992

15. Khoshafian, S. *Object-Orienled Databases*. Jolm Wiley & Sons, Inc. 1993.

16. Kim H. *An Algorithmic and Computational A.5pects ofOODB SchemaDesign* en R.Gupta y E.Horowitz (eds.) Object-Oriented Databases with Applications to Case, Network and VLSI CAD. Prentice-Hall Series in Data and Knowledge Base System, 1991.

17. Kim W. *Introduction lo Object-Oriented Databases*. The MIT Press, 1990.

18. Kim W. et al. *Architecture oflhe Orion Next- Generation Database System* en M.Stonebraker Readings in Database Systems. Morgan Kaufman Publishers, 1994.

19. Korth H. y Silberschatz A. *Fundamentos de Bases de Datos,* McGraw-Hill, 1995.

20. Lamb C. et al. *The ObjeclStore Database System* en M.Stonebraker Readings in Database Systems. Margan Kaufman Publishers, 1994.

21. Lecuse C. et al. *An Object-Orienled Data Model* en F.Bancilhon, C. Delobel y P. Kanelakis (eds.) Building and Object-Oriented Database System. The Story of O_2. Morgan Kaufman Publishers, 1992

22. Martin B. y Pederson C.H. *Long Living Concurrenl Activities* en M.T. Ozsu, U. Dayal y P. Valduriez (eds.) Distributed Object Management. Morgan Kaufman Publishers, 1994.

23. Masunga *YObject Identity. Equalityand Relational Concepls* en W.Kim, 1.M.Nicolas y S.Nishio (eds.) Deductive and Object-Oriented Databases. North-Holland, 1991

24. Ozsu M.T. y Valduriez P *PrincipIes oI Distributed Database Systems*. Prentice Hall, 1991.

25. Rumbaugh J et al. *A10delado y Diseño Orienlados a Objelos*. Prentice Hall, 1991

26. Sholl M.H. et al. *Object Algebra and Views forMulti-Objectbases* en M.T. Ozsu, U. Dayal y P.Valduriez (eds.) Distributed Object Management. Morgan Kaufman Publishers, 1994.

27. Sommerville I. *Software Engineering*. Addison-Wesley, 1992.

28. Tanaka K. y ChangT.S. *On Nalural Join in Object-Oriented Databases* en W.Kim, 1.M.Nicolas y S.Nishio (eds.) Deductive and Object-Oriented Databases. North-Holland, 1991

29. Teorey T.J. et al. *A Logical Design Methodology for Relational Dafabases Using the Extended Entity-RElationship Model.* ACM Computing Surveys, 18(2), 1986.

30. Yourdon, E., *Object-orienled Analysis.* Prentice-Hall, 1990.