TEMA 50

ANÁLISIS DE SISTEMAS: MODELIZACIÓN CONCEPTUAL DE DATOS. TÉCNICAS DESCRIPTIVAS. DOCUMENTACIÓN.

ÍNDICE

1. INTRODUCCIÓN

En este capítulo se explora una notación gráfica para modelar datos. El **diagrama de entidad-relación** (también conocido como **DER**, o **diagrama E-R**) es un modelo de red que describe con un alto nivel de abstracción la distribución de datos almacenados en un sistema. Es muy diferente del DFD, que modela las funciones que lleva a cabo un sistema; y es diferente del diagrama de transición de estados, que modela el comportamiento dependiente del tiempo de un sistema.

Para el analista, el DER representa un gran beneficio: enfatiza las relaciones entre almacenes de datos en el DFD que de otra forma se hubieran visto sólo en la especificación de proceso. Cada una de las cajas rectangulares corresponde a un almacén de datos en un DFD, y puede verse que hay relaciones (conexiones) que normalmente no se aprecian en un DFD. Esto se debe a que el DFD enfoca la atención del lector a las funciones que el sistema efectúa, no a los datos que ocupa.

2. LOS COMPONENTES DE UN DIAGRAMA E-R

Hay cuatro componentes principales en un diagrama de entidad-relación:

- Tipos de objetos.
- Relaciones.
- Indicadores asociativos de tipo de objeto.
- Indicadores de supertipo/subtipo.

2.1. Tipos de objetos

El tipo de objeto se representa en un diagrama de entidad-relación por medio de una caja rectangular; en la figura se muestra un ejemplo.

Figura 50.1. Un tipo de objeto

Representa una colección o conjunto de objetos (cosas) del mundo real cuyos miembros individuales (o instancias) tienen las siguientes características:

> *Cada uno puede identificarse de manera única por algún medio*. Existe alguna forma de diferenciar entre instancias individuales del tipo de objeto. Por ejemplo, si se tiene un tipo de objeto conocido como cliente, debemos ser capaces de distinguir uno de otro (tal vez por un número de cuenta, por su apellido, o por su número de Seguro Social). Si todos los clientes son iguales (si hay un negocio en el que son sólo entes sin cara y sin nombre que entran a la tienda a comprar cosas), entonces cliente no sería un tipo de objeto con significado.
>
> *Cada uno juega un papel necesario en el sistema que se construye*. Es decir, para que el tipo de objeto sea legítimo, debe poder decirse que el sistema no puede operar sin tener acceso a esos miembros. Si se está construyendo un sistema de ingreso de pedidos para la tienda, por ejemplo, se pudiera pensar que, además de los clientes, la tienda tiene mozos, cada uno de los cuales se identifica de manera individual por su nombre. A pesar de que los mozos juegan un papel útil en la tienda, el sistema de ingreso de pedidos puede funcionar felizmente sin ellos; por tanto, no merecen un papel como tipos de objeto en el modelo del sistema. Obviamente, esto es algo que debe verificarse con los usuarios al construir el modelo.

Cada uno puede describirse por uno o más datos. Es decir, un cliente puede describirse por medio de datos tales como nombre, domicilio, límite de crédito y número telefónico. Muchos textos sobre bases de datos describen esto como "asignar datos a un tipo de objeto". Nótese que los atributos deben aplicarse a cada instancia del tipo de objeto; por ejemplo, cada cliente debe tener nombre, domicilio, límite de crédito, número telefónico etc.

En muchos de los sistemas que desarrolle, los tipos de objetos serán la representación en el sistema de algo material del mundo real. Esto significa que los clientes, artículos de inventario, empleados, partes manufacturadas, etc., son objetos típicos. El objeto es el algo material del mundo real, y el tipo de objeto es su representación en el sistema. Sin embargo, un objeto también pudiera ser algo no material: por ejemplo, horarios, planes, estándares, estrategias y mapas.

Dado que a menudo las personas son tipos de objetos en un sistema, debe tenerse otra cosa en mente: una persona (o para el caso, cualquier cosa material) pudiera ser diversos tipos de objetos distintos en distintos modelos de datos, o incluso en un mismo modelo. Juan Pérez, por ejemplo, puede ser, empleado en un modelo de datos y cliente en otro. También pudiera ser empleado y cliente dentro del mismo modelo.

Nótese que en todos los ejemplos de un objeto se ha usado un sustantivo singular (por ejemplo, empleado, cliente). Esto no es necesario, pero es un convenio útil; existe una correspondencia entre objetos en el DER y almacenes en el DFD; así, si existe un objeto cliente en el DER, debe haber un almacén de clientes en el DFD.

2.2. Relaciones

Los objetos se conectan entre sí mediante relaciones. Una relación representa un conjunto de conexiones entre objetos, y se representa por medio de un rombo. La figura muestra una relación sencilla que pudiera existir entre dos o más objetos.

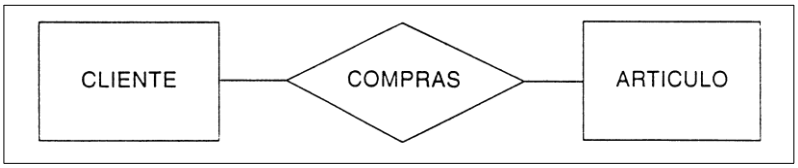

Figura 50.2. Una relación

Es importante reconocer que la relación representa un conjunto de conexiones. Cada instancia de la relación representa una asociación entre cero o más ocurrencias de un objeto y cero o más ocurrencias del otro. Así, en la figura, la relación etiquetada como compras puede contener las siguientes instancias individuales:

- Instancia 1: el cliente 1 compra el artículo 1.
- Instancia 2: el cliente 2 compra los artículos 2 y 3.
- Instancia 3: el cliente 3 compra el artículo 4.
- Instancia 4: el cliente 4 compra los artículos 5, 6 y 7.
- Instancia 5: el cliente 5 no compra ningún artículo.
- Instancia 6: los clientes 6 y 7 compran el artículo 8.
- Instancia 7: los clientes 8, 9 y 1 0 compran los artículos 9, 10 y 11.
- Etc.

Como puede verse, entonces, una relación puede conectar dos o más instancias del mismo objeto.

Nótese que la relación representa algo que debe ser recordado por el sistema: algo que no pudo haberse calculado ni derivado mecánicamente. Así, el modelo de datos de la figura indica que existe alguna razón relacionada con el usuario para recordar el hecho de que el cliente 1 compra el artículo 1, etc. Y la relación también indica que no existe nada a priori que hubiera permitido determinar que el cliente 1 compró el artículo 1 y nada más. La relación representa la memoria del sistema (un objeto representa la memoria del sistema también, desde luego).

Nótese también que puede existir más de una relación entre dos objetos. La figura siguiente, por ejemplo, muestra dos relaciones distintas entre un paciente y un médico. A primera vista, pudiera pensarse que esto es rebuscar lo obvio: cada vez que el médico trata a un paciente, le cobra mediante un recibo. Pero la figura sugiere que la situación puede ser distinta; pudiera resultar, por ejemplo, que existan diversas instancias de "tratamiento" entre un médico y el mismo paciente (es decir, un tratamiento inicial, tratamientos de seguimiento, etc.). La figura implica que la relación de recibos está completamente separada de la relación de tratamiento: tal vez a algunos pacientes se les dan recibos sólo por su primer tratamiento, mientras que a otros se les dan para cada tratamiento y a otros jamás se les dan.

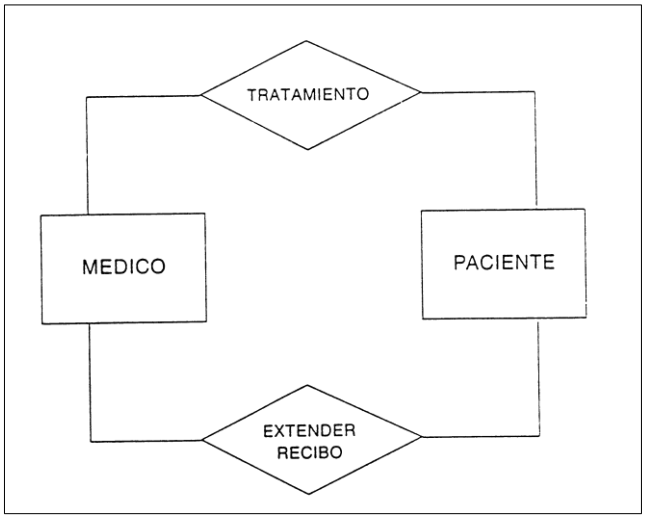

Figura 50.3. (a). Relaciones múltiples entre objetos

Una situación más común es ver múltiples relaciones entre múltiples objetos. La figura 50.3. (b) muestra la relación que existe típicamente entre un cliente, un vendedor, un agente de bienes raíces, el abogado del cliente y el abogado del vendedor, para la compra-venta de una casa.

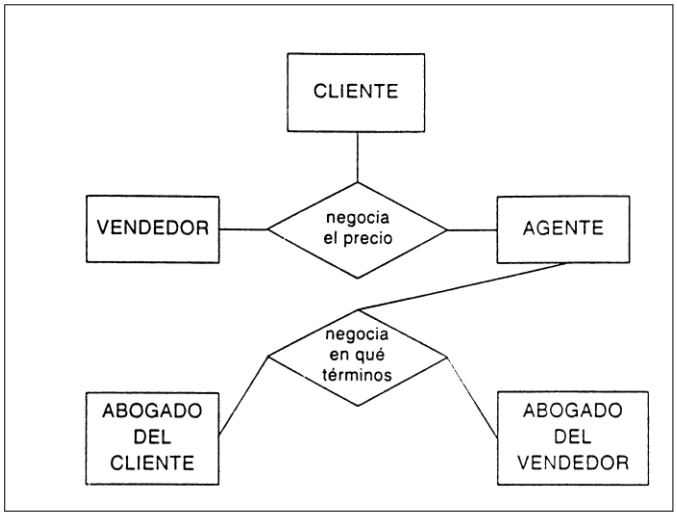

Figura 50.3. (b)

Con un diagrama complejo como el de la figura 50.3. (b) (que es típico, y tal vez más simple que los DER que es probable encontrar en un proyecto real), la relación y sus tipos de objetos deben leerse como una unidad. La relación se puede describir desde la perspectiva de cualquiera de los tipos de objetos participantes, y todas esas perspectivas son válidas. De hecho, el conjunto de todos estos puntos de vista es el que describe completamente la relación. Por ejemplo, en la figura puede verse la relación de negociación de precios en cualquiera de las siguientes tres formas:

1. El agente de bienes raíces negocia el precio entre el cliente y el vendedor.

2. El cliente negocia el precio con el vendedor, mediante el agente de bienes raíces.

3. El vendedor negocia el precio con el cliente, mediante el agente de bienes raíces.

Nótese que, en algunos casos, puede haber relaciones entre diferentes instancias de un mismo tipo de objeto. Por ejemplo, imagine un sistema que se esté desarrollando para una universidad, en el cual curso, estudiante y profesor son tipos de objetos. La mayoría de las relaciones de interés serán entre instancias de diferentes tipos de objetos (por ejemplo, las relaciones "se inscribe en", "imparte", etc.). Sin embargo, pudiera requerirse modelar la relación "es prerrequisito para" entre una instancia de curso y otra.

3. REGLAS PARA LA CONSTRUCCIÓN DE DIAGRAMAS ENTIDAD-RELACIÓN

La notación que se muestra en la sección anterior es suficiente para construir diagramas E-R arbitrariamente complejos. Sin embargo, podría estar pensando en este momento: "¿Cómo descubrir qué son, para comenzar, los objetos y las relaciones?". El *modelo inicial de objetos y relaciones* usualmente se derivará de **1)** su comprensión de la aplicación del usuario, **2)** entrevistas con el usuario y **3)** cualquier otro tipo de investigación y recolección de información que pueda usar.

No espere que el primer diagrama E-R que haga sea el final, que revisará con la comunidad usuaria o que entregará a los diseñadores del sistema. Como los diagramas de flujo de datos y todas las demás herramientas de modelado, los diagramas E-R deben revisarse y mejorarse muchas veces; la primera versión típicamente no será más que un borrador, y las versiones subsecuentes se producirán utilizando una serie de reglas de refinamiento que se presentan en esta sección. Algunas de las reglas de refinamiento llevan a la creación de tipos adicionales de objeto, mientras que otras llevarán a la eliminación de objetos y/o relaciones.

3.1. Añadir tipos de objetos adicionales

Como se indicó anteriormente, el primer diagrama E-R típicamente se creará a partir de entrevistas iniciales con el usuario, y de su conocimiento de la materia en cuanto al negocio del usuario. Esto, desde luego, le dará una buena pista respecto a la identidad de los principales objetos y relaciones.

Después de haber desarrollado el primer DER, el siguiente paso es *asignar los datos del sistema a los diversos tipos de objetos*. Se supone, desde luego, que sabe cuáles son los datos. Esto puede suceder en cualquiera de tres maneras:

1. Si el modelo del proceso (el DFD) ya se ha desarrollado o se está desarrollando paralelamente al modelo de datos, entonces el diccionario de datos ya existirá. Pudiera no estar completo aún, pero lo que haya será suficiente para comenzar el proceso de asignación.

2. Si el modelo del proceso no se ha desarrollado (o en el caso extremo, si no tiene intención de desarrollar uno), entonces pudiera tener que empezar por entrevistar a todos los usuarios apropiados para construir una lista exhaustiva de datos (y sus definiciones). Al hacer esto, puede asignar los datos a los objetos en el diagrama de E-R. (Sin embargo, note que esto es un proceso ascendente que consume tiempo, y que pudiera ocasionar retrasos y frustración.).

3. Si está trabajando con un grupo activo de administración de datos, hay una buena probabilidad de que ya exista un diccionario de datos, que podría obtenerse pronto durante el proyecto, de manera que en ese momento ya pudiera comenzar el proceso de asignación.

El proceso de asignación puede ofrecer una de tres *razones para crear nuevos tipos de objetos*:

1. Es posible descubrir datos que se pueden asignar a algunas instancias de un tipo de objeto pero no a otras.
2. Pudieran descubrirse datos aplicables a todas las instancias de dos objetos distintos.
3. Podría descubrirse que algunos datos describen relaciones entre otros tipos de objetos.

Si durante el proceso de asignar datos a tipos de objetos encuentra que algunos datos no se pueden aplicar a todas las instancias de algún tipo de objeto dado, necesitará crear un conjunto de **subtipos** bajo el tipo de objeto con el que ha estado trabajando, y asignar los datos específicos a los subtipos apropiados.

Suponga que, por ejemplo, se está desarrollando un sistema de personal, y se ha identificado (con gran perspicacia y creatividad) un tipo de objeto llamado empleado. Al revisar los datos disponibles se encuentra que muchos de ellos (edad, estatura, fecha de contratación, etc.) se aplican a todas las instancias de un empleado. Sin embargo, luego se descubre un dato llamado número-de-embarazos; se trata obviamente de un dato relevante para las empleadas, pero no para los empleados varones. Esto llevaría a crear empleados-varones y empleadas como subtipos de la categoría general de empleado.

Obviamente, no se está sugiriendo que todos los sistemas de personal deban hacer seguimiento del número de embarazos que cada empleada ha tenido; el ejemplo se escogió meramente porque existe un consenso general de que los empleados varones no se pueden embarazar. Compare esto, sin embargo, con el dato nombre-del-cónyuge: hay varias instancias de empleado para quienes no se aplicaría esto (porque son solteros), pero es una situación muy distinta a la de un dato que no se puede aplicar definitivamente.

En la mayoría de los casos el proceso de crear nuevos subtipos y asignarles datos de manera apropiada es bastante directo. Sin embargo, debe tenerse siempre en mente una situación excepcional: pudiera suceder que todos los datos relevantes se atribuyan a uno de los subtipos, y que ninguno de los datos se pueda asignar al objeto supertipo; es decir, puede suceder que los datos sean mutuamente excluyentes, perteneciendo a un subtipo o a otro pero no a ambos. Suponga, por ejemplo, que los únicos datos que se puede asignar a los empleados son número-de-embarazos y años-de-experiencia-jugando-con-el-equipo-de-baloncesto. Podría decidirse (tras preguntarnos a qué lunático usuario pudiera habérsele ocurrido tal sistema) que el supertipo general empleado no se aplica.

También puede ocurrir la situación inversa: los datos pueden describir instancias de dos (o más) tipos distintos de objetos de la misma manera. Si esto ocurre, debe crearse un **supertipo** nuevo y asignarle los datos comunes al supertipo. Por ejemplo, tal vez se identificó cliente-de-contado y cliente-a-crédito como dos tipos de objetos distintos cuando se creó el DER para un sistema de pedidos (tal vez porque el usuario señaló que eran dos categorías distintas). Sin embargo, pronto se hace evidente que los datos nombre-del-cliente y domicilio-del-cliente describen ambos tipos de cliente de la misma forma, lo cual apoyaría la creación de un supertipo.

Similarmente, si un dato describe la interacción de dos o más tipos de objetos, entonces debería reemplazarse la relación "desnuda" entre los dos objetos con un tipo asociativo de objeto. Por ejemplo, en el primer borrador de DER podría existir una relación de compra entre cliente y artículo. Durante la asignación de datos pudiera encontrarse con que hay un dato llamado fecha-de-compra que 1) parece pertenecer a la relación compra y 2) obviamente describe, o proporciona datos acerca de la interacción de un cliente con un artículo. Esto sugiere que debe sustituirse la relación compra por un tipo asociativo de objeto.

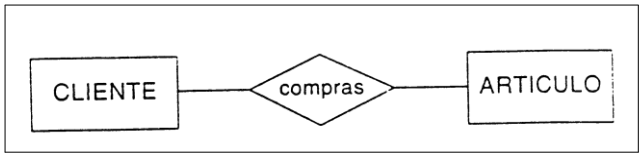

Figura 50.4. Diagrama E-R inicial

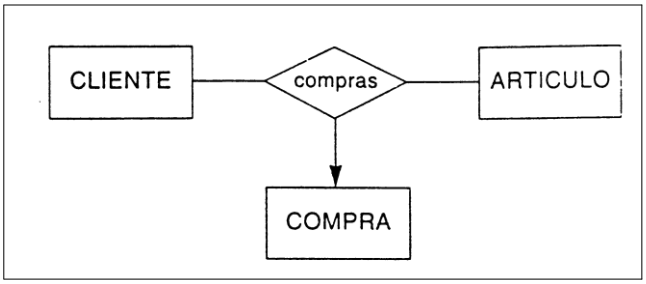

Figura 50.5. Reemplazo de una relación por un tipo asociativo de objeto

A veces el diagrama E-R inicial tendrá un tipo de objeto que, visto de cerca, claramente parece ser un tipo asociativo de objeto. Por ejemplo, la figura 50.6 muestra un diagrama E-R con tres objetos relacionados: cliente, pedido y producto. Durante el proceso de asignar datos a los diversos objetos, se encuentra que fecha-de-entrega en realidad se aplica al objeto pedido porque a los clientes no se les entrega en ningún lado, y los productos se entregan sólo como resultado de un pedido. De hecho, esto hace evidente que pedido en sí es una relación entre cliente y producto, además de un objeto acerca del cual interesa recordar algunos hechos. Esto lleva a la figura 50.7.

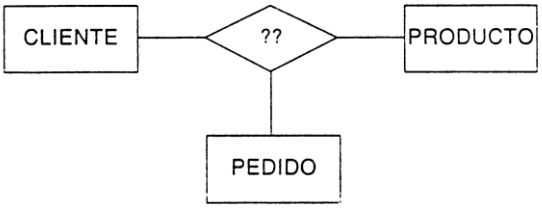

Figura 50.6. DER inicial

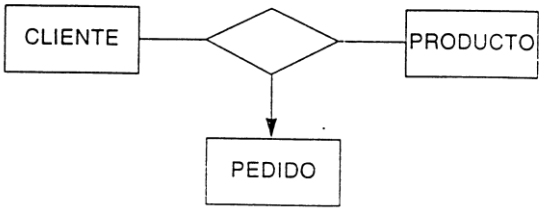

Figura 50.7. Un objeto transformado en objeto asociativo

Finalmente, tenemos el caso de grupos que se repiten. Considere, por ejemplo, el tipo de objeto empleado, con los datos obvios como nombre y domicilio. Suponga que hay datos adicionales como nombre-del-hijo, edad-del-hijo y sexo-del-hijo. Podría argumentarse obviamente que son formas de describir un objeto nuevo llamado hijo, que inadvertidamente se había incluido anteriormente en empleado. Podría también argumentarse que existen (potencialmente) múltiples instancias de información relacionadas con hijos en cada instancia de un empleado, y que cada instancia de información relacionada con los hijos se define de manera única por el nombre-del-hijo. En este caso, el tipo de objeto que inicialmente se imaginó debe transformarse en dos objetos tipo, conectados por una nueva relación, como se muestra en la figura 50.8.

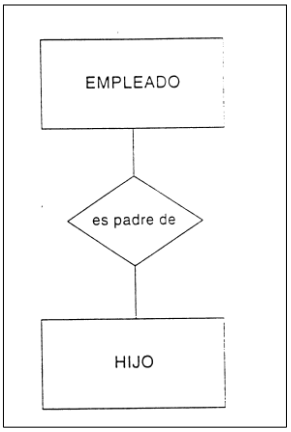

Figura 50.8. Diagrama E-R revisado

Este proceso de eliminar objetos incluidos en otros es parte de una actividad de refinamiento más general llamada normalización. El objetivo de la normalización es producir tipos de objetos, en los que cada instancia (o miembro) consiste en un valor llave primario que identifica a alguna entidad, junto con un conjunto de valores de atributo independientes que describen a la entidad de alguna manera.

3.2. Eliminar tipos de objetos

La sección anterior trató los refinamientos del DER que crean objetos y/o relaciones adicionales. Sin embargo, existe un buen número de situaciones en las que los refinamientos del DER llevan a la eliminación de tipos de objetos y relaciones redundantes o erróneas. Examinaremos cuatro situaciones comunes:

1. Tipos de objetos que consisten sólo en un identificador.
2. Tipos de objetos para los cuales existe una sola instancia.
3. Tipos asociativos de objetos flotantes.
4. Relaciones derivadas.

Si se tiene un diagrama E-R en el cual uno de los tipos de objeto tiene sólo un identificador asignado como dato, existe la oportunidad de eliminar el tipo de objeto y asignar el identificador, como dato, a un tipo de objeto relacionado. Por ejemplo, imagine que se construyó un DER como muestra la figura para un sistema de personal:

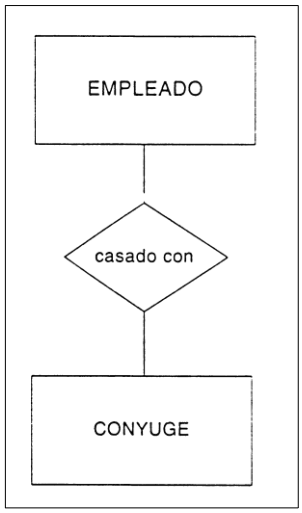

Figura 50.9. Diagrama E-R inicial

Durante el proceso de asignar datos a los diversos objetos, sin embargo, podría encontrarse que la única información que el sistema mantiene acerca del cónyuge es su nombre (es decir, el identificador que distingue a uno de cualquier otro en el sistema). En este caso, un refinamiento obvio sería eliminar cónyuge como tipo de objeto e incluir nombre-del-cónyuge como dato dentro del objeto empleado.

Observe que este refinamiento sólo tiene sentido si existe una correspondencia uno a uno entre instancias del objeto que está a punto de ser eliminado e instancias del objeto relacionado. El ejemplo anterior tiene sentido porque la sociedad moderna supone que una persona tendrá cuando más un cónyuge. Esto lleva al diagrama E-R reducido que se muestra en la figura.

Figura 50.10. Diagrama E-R reducido

Se puede hacer una reducción aún mayor si encontramos que el diagrama E-R inicial contiene un objeto para el cual el único hecho es el identificador, y ése es un objeto de una sola instancia. Considere el diagrama E-R inicial que se muestra en la figura.

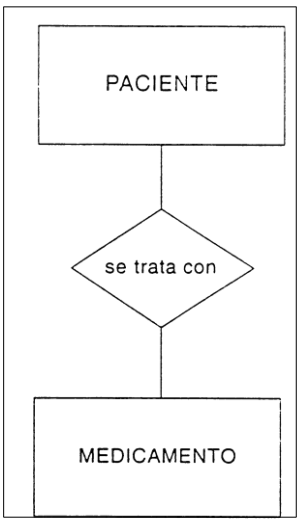

Figura 50.11. Diagrama E-R inicial

A primera vista parece ser una manera razonable de mostrar la relación entre pacientes y medicinas en un hospital. Pero suponga que la única información que se guarda acerca del medicamento es su nombre (identificador); y suponga que el hospital sólo administra un tipo de medicamento (por ejemplo, aspirina). En este caso, el medicamento es una constante y ni siquiera tiene que mostrarse en el diagrama. (Note que esto también significa que el sistema no tendría un almacén de datos llamado medicamentos.) El diagrama reducido se vería como la figura 12.

Figura 50.12. Diagrama E-R reducido

Debido a la situación anterior, es posible crear un tipo asociativo de objeto flotante. Considere la siguiente variante del ejemplo del hospital anterior, que muestra la figura 13. Si como se sugirió anteriormente, resulta que medicamento es un objeto de instancia única, sólo con identificador, entonces se eliminaría. Esto resultaría en el diagrama reducido de la figura 14; nótese que tratamiento todavía es un tipo de objeto asociativo, aunque se conecte sólo con un tipo de objeto. Esto se conoce como tipo de objeto asociativo flotante y es legal (aunque poco usual) en un DER.

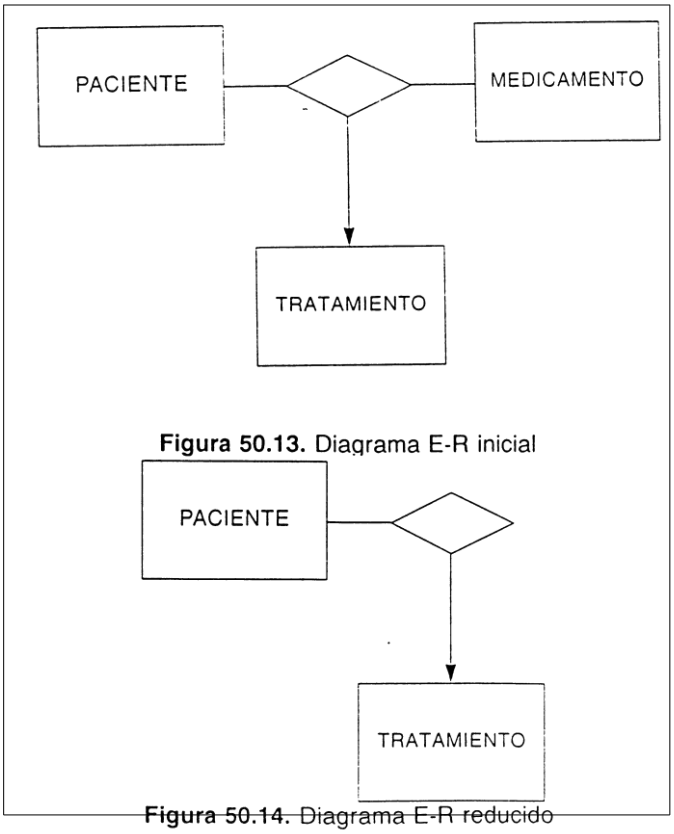

Figura 50.13. Diagrama E-R inicial

Figura 50.14. Diagrama E-R reducido

Finalmente, las relaciones que se pueden derivar, o calcular, deben eliminarse del diagrama E-R inicial. Como se mencionó anteriormente en este capítulo, el DER debe mostrar los requerimientos para los datos almacenados. Por ello, en la figura 15, si la relación renovar entre conductor y licencia se puede derivar (basándose en el cumpleaños del conductor, o en la primera letra de su apellido, o en algún otro esquema usado en la oficina de tránsito), entonces debe eliminarse. Esto lleva a la figura 16, en la cual los tipos de objeto no están conectados. Esto es legal en un DER; no es necesario que todos los tipos de objetos estén conectados entre sí.

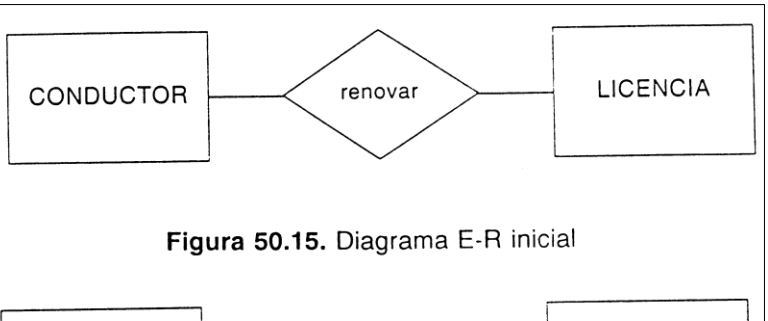

Figura 50.15. Diagrama E-R inicial

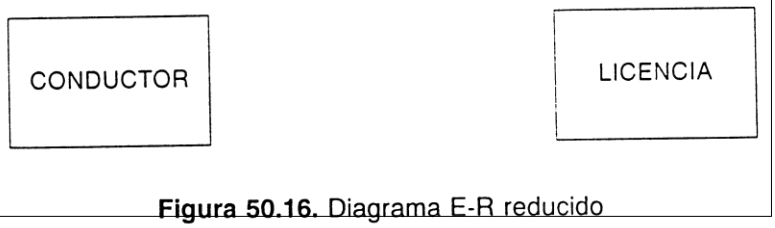

Figura 50.16. Diagrama E-R reducido

4. EXTENSIONES AL DICCIONARIO DE DATOS PARA DIAGRAMAS E-R

Finalmente, observamos que el diccionario de datos necesita extenderse para tomar en cuenta la notación de Diagrama E-R. En general, los objetos del DER corresponden con almacenes del DFD. Esto significa que en la definición sacada del diccionario de datos que se da a continuación, cliente es tanto definición del tipo de objeto como instancia del almacén clientes.

CLIENTES = {CLIENTE}

Cliente = @nombre-del-cliente + domicilio + número-telefónico

Nótese también que la definición de un cliente incluye la especificación del campo llave, que es el dato (o atributo) que diferencia una instancia de un cliente de cualquier otra. El signo de arriba (@) se utiliza para indicar el o los campos llave.

Sin embargo, también hay que incluir en el diccionario de datos una definición de todas las relaciones que se muestran en el DER. La definición de relación debe incluir una descripción de su significado en el contexto de la aplicación; y debe indicar los objetos que forman la asociación. Los límites superiores e inferiores apropiados deben especificarse para indicar si la asociación es de *uno a uno, de uno a muchos o de muchos a muchos*. Por ejemplo, la relación compras que se muestra en la figura 50.4. puede definirse en el diccionario de datos de la siguiente forma:

compras = *la asociación de un cliente y uno o más artículos*

@identidad-del-cliente + {@identidad-del-artículo + cantidad-comprada}

5. BIBLIOGRAFÍA

Pressman, Roger S.
Ingeniería del Software
Mc Graw-Hill, 1993

López, Antonio
Metodologías de Desarrollo
Ra-ma, 1990