GENERACIÓN AUTOMÁTICA DEL DIAGRAMA ENTIDAD-RELACIÓN Y SU REPRESENTACIÓN EN SQL DESDE UN LENGUAJE CONTROLADO (UN-LENCEP)

Carlos Mario Zapata Jaramillo*
Guillermo González Calderón**
John Jairo Chaverra Mojica***

Recibido: 08/03/2011
Aceptado: 11/05/2011

RESUMEN

Entidad-relación es uno de los diagramas que se utilizan en el desarrollo de modelos para representar la información de un dominio. Con el fin de agilizar y mejorar el proceso de desarrollo de software, diferentes propuestas surgieron para contribuir en la obtención automática o semiautomática del diagrama entidad-relación. Varias de estas propuestas utilizan como punto de partida lenguaje natural o lenguaje controlado, mientras otras propuestas utilizan representaciones intermedias. Los interesados en el desarrollo de una aplicación de software no suelen comprender varias de las representaciones utilizadas sin tener previa capacitación, lo cual restringe la participación activa del interesado en todas las etapas del desarrollo. Con el fin de solucionar estos problemas, en este artículo se propone un conjunto de reglas heurísticas para la obtención automática del diagrama entidad-relación y su representación en SQL. Se toma como punto de partida el lenguaje controlado UN-Lencep, que ya se emplea para la generación de otros artefactos en el desarrollo de aplicaciones de software.

Palabras clave: UN-Lencep, entidad-relación, SQL

* Ingeniero civil, Ph D en Ingeniería. Profesor asociado de la Universidad Nacional de Colombia, Líder del grupo de investigación en Lenguajes Computacionales. Correo electrónico cmzapata@unal.edu.co. Teléfono: (57)(4) 4255374. Fax: (57)(4) 4255365 Carrera 80 No. 65-223 Bloque M8A-310. Facultad de Minas, Escuela de Sistemas.

** Msc. en Ingeniería de Sistemas. Doctor(C) en Ingeniería. Grupo de investigación ARKADIUS. Docente Ingeniería de Sistemas. Universidad de Medellín. Correo electrónico: ggonzalezc@udem.edu.co.

*** Ingeniero de sistemas e informática. Msc. en Ingenieríade Sistemas. Universidad Nacional de Colombia. Correo electrónico jjchaver@unal.edu.co.

AUTOMATIC GENERATION OF ENTITY-RELATIONSHIP DIAGRAM AND ITS REPRESENTATION IN SQL FROM A CONTROLLED LANGUAGE (UN-LENCEP)

ABSTRACT

Entity-relationship diagram (ERD) is one of the used in modelling the domain information. Several proposals have emerged for speeding up and improving the software development process by either automatically or semi-automatically obtain the ERD. Natural language, controlled languages, and intermediate representations have been used in such a task. The stakeholders (people with some concern in application development), when untrained, are incapable to understand several of such representations. As a consequence, stakeholder active participation in software development is highly restricted. Trying to solve these problems, a set of heuristic rules for automatically obtaining ERD and its SQL-based equivalence is proposed in this paper. The starting point is UN-Lencep, a controlled language already used for generating other artefacts belonging to software application development.

Key words: UN-Lencep, entity relationship, SQL.

INTRODUCCIÓN

Entidad-Relación [1] es un diagrama de ingeniería de software que se utiliza para desarrollar un modelo de datos de alta calidad. Este diagrama, paulatinamente se está convirtiendo en la técnica universal para modelar datos. Por ello, con el fin de mejorar y agilizar el proceso de desarrollo de software, son diversas las propuestas dirigidas a permitir la obtención automática y semiautomática de los diferentes elementos del diagrama entidad–relación y su correspondiente representación en el lenguaje SQL (*Structured Query Language*).

Algunos autores [2-5] proponen la obtención semiautomática del diagrama entidad-relación a partir de lenguaje natural. Para tal fin, proponen un conjunto de reglas heurísticas que se basan, principalmente, en la sintaxis de las frases. Estas propuestas se concentran en la obtención de las entidades y sus atributos. En acciones más complejas, como es la definición de relaciones entre las entidades, es necesaria la intervención del analista para resolver las ambigüedades.

Siguiendo la misma tendencia, Gangopadhyay [6] propuso la obtención automática del diagrama entidad-relación a partir de un lenguaje controlado. Esta propuesta utiliza un diagrama de dependencias conceptuales como representación intermedia. Al utilizar representaciones intermedias, se ve restringida la participación activa del interesado en el proceso de desarrollo del software.

Con el fin de dar solución a estos problemas, en este artículo se propone un conjunto de reglas heurísticas para obtener, automáticamente, el diagrama entidad-relación y su correspondiente representación en SQL a partir de un lenguaje controlado, UN-Lencep (Universidad Nacional de Colombia—Lenguaje Controlado para la Especificación de Esquemas Preconceptuales). Al utilizar UN-Lencep se mejora la comunicación entre el analista e interesado, ya que UN-Lencep es de fácil comprensión por su cercanía con el lenguaje natural.

Este artículo se organiza de la siguiente manera: en la sección 2 se define el marco teórico que agrupa los conceptos de este dominio; en la sección 3 se resumen algunos trabajos en obtención automática y semiautomática del diagrama entidad-relación; en la sección 4 se plantea un conjunto de reglas heurísticas para la obtención automática del diagrama entidad-relación y su representación en SQL; en la sección 5 se plantea un caso de estudio para ejemplificar el uso de la reglas; las conclusiones y el trabajo futuro se incluyen en la sección 6.

1 MARCO TEÓRICO

1.1 UN-LENCEP Un lenguaje controlado para obtención automática de diagramas UML

La educción de requisitos es una de las primeras fases en el desarrollo de software. En ella, el interesado suele expresar el dominio de un problema. En este discurso es posible identificar artefactos que constituyen parte fundamental de la solución del problema. UN-Lencep (Universidad Nacional de Colombia—Lenguaje Controlado para la Especificación de Esquemas Preconceptuales) se diseñó, inicialmente, para que el interesado pudiera expresar las ideas de un dominio específico, con el fin de realizar su traducción automática hacia los esquemas preconceptuales. En la tabla 1 se presentan las equivalencias de sus especificaciones básicas en un subconjunto del lenguaje natural [7].

Tabla 1. Equivalencias de UN-Lencep en un subconjunto del lenguaje natural

Construcción Formal	*Expresión en Lenguaje Natural Controlado*
A <ES> B	A *es una clase de* B
A <TIENE> B	B pertenece a A B es una parte de A B está incluido en A B está contenido en A B es un elemento de A B es un subconjunto de A

Construcción Formal	*Expresión en Lenguaje Natural Controlado*
A <R1> B	<R1> puede ser cualquier verbo dinámico, por ejemplo: A registra B, A paga B
C <R2> D, <SI> A <R1> B	si A <R1> B entonces C <R2> D Dado que A <R1> B, C <R2> D Luego de que A <R1> B, C <R2> D
<SI> {COND}<ENTONCES> A <R1> B, <SINO> C <R2> D	{COND} es una condición expresada en términos de conceptos. <R1> y <R2> son verbos dinámicos. <SINO> es opcional, por ejemplo: si M es mayor que 100 entonces A registra B

Fuente: elaboración propia.

2 GENERACIÓN AUTOMÁTICA DEL DIAGRAMA ENTIDAD–RELACIÓN

RADD (*Rapid Application and Database Development*) [2], *ER-Converter* [3], *E-R Generator* [4] y ANNAPURNA [5] procuran elaborar un diagrama entidad-relación a partir de especificaciones en lenguaje natural y, luego, promover la completitud del diagrama obtenido mediante un diálogo controlado con el usuario. En *ER-Converter* se utilizan dos tipos de reglas: normas vinculadas a la semántica y normas genéricas que identifican entidades y sus atributos. En *E-R Generator,* en algunos casos, es necesario que el analista intervenga para resolver las ambigüedades tales como la fijación de los atributos y las relaciones con las demás entidades. ANNAPURNA define un conjunto de reglas semánticas con el fin de extraer las entidades y sus relaciones. Luego, estas entidades se representan en *S-diagram*, que es un modelo de datos gráfico utilizado para especificar las entidades, atributos y las conexiones entre entidades. *S-diagram* funciona mejor cuando la complejidad es pequeña.

Gangopadhyay [6] continuó con esta tendencia proponiendo una herramienta CASE (*Computer Aided Software Engineering*) para la obtención automática del diagrama entidad-relación a partir de un lenguaje controlado. Esta herramienta emplea un diagrama de dependencias conceptuales como representación intermedia a partir del lenguaje controlado, y un *parser* basado en una red de transición aumentada para el procesamiento de las diferentes palabras. Esta propuesta se implementó en un prototipo usando Oracle como gestor de bases de datos.

TRIDENT(*Tools foR an Interactive Development EnvironmeNT*) [8-11], emplea análisis de requisitos funcionales y análisis de tareas. A partir del análisis de los requisitos funcionales se obtiene un diagrama entidad-relación extendido. El análisis de tareas se obtiene construyendo un ACG (*Activity Chan Graphs*) que liga las tareas interactivas del usuario con la funcionalidad del sistema.

La mayoría de estas propuestas son procesos semiautomáticos y, en muchos casos, requieren la intervención del analista para resolver ambigüedades, como es el caso de *E-R Generator* y ANNAPURNA. En el caso de ANNAPURNA, el problema es aún mayor, puesto que en modelos grandes tiende a fallar. Algunas propuestas utilizan diagramas intermedios para lograr el objetivo final, lo cual tiende a generar problemas de comunicación con el interesado, debido a su complejidad.

3 OBTENCIÓN AUTOMÁTICA DEL DIAGRAMA ENTIDAD RELACIÓN Y SU REPRESENTACIÓN EN SQL, A PARTIR DE UN LENGUAJE CONTROLADO (UN-LENCEP)

UN-Lencep posee un conjunto básico de plantillas para facilitar su uso a los interesados. Los principales componentes de UN-Lencep son: [7].

- **A <ES> B:** Representa una relación de generalización, en la cual el concepto de origen es el subtipo y el concepto destino el super-tipo.

- **A <TIENE> B:** El concepto destino se representa como un atributo del concepto, siempre y cuando el concepto destino no se identifique, en sí mismo, como una entidad.
- **A <R1> B:** El concepto origen ejecuta una operación (R1) sobre el concepto destino. R1 es una operación del concepto destino.
- **C <R2> D, <SI> A <R1> B**: En la ejecución de la primera operación, el concepto origen C ejecuta una operación sobre el concepto destino D, siempre que ocurra, que el concepto destino A ejecute una operación sobre el concepto destino B. Así, la primera operación mencionada se invocará después de realizar el proceso que establece la operación entre A y B.
- **<SI> {COND} <ENTONCES> A <R1> B, <SINO> C <R2> D**: La operación R1 que A realiza sobre B, se ejecuta sólo si COND se cumple; en caso de que no se cumpla, entonces C realizará la acción R2 sobre B. La ejecución del condicional se lleva a cabo dentro de una operación E <R3> F, donde F es una entidad que tiene como atributo algún elemento que aparece en el condicional.

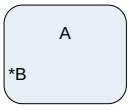
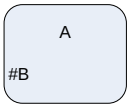
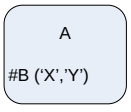
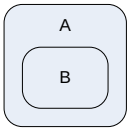
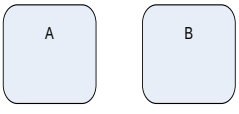
UN-Lencep aún no posee artefactos que permitan identificar los tipos de datos en un discurso. Por ello, en el marco de este trabajo las claves primarias y las claves foráneas se definirán de tipo "*int*" y los demás atributos serán de tipo "*varchar*". Para cada tabla se definirá como clave principal un "*id*"autonumérico.

Las reglas para la traducción automática de UN-Lencep al diagrama entidad-relación se clasifican en dos tipos; aquellasque tienen como precondición elementos propios de UN-Lencep (tipo A) y, las que tienen como precondición, además de los elementos propios de UN-Lencep, elementos del diagrama de clases (tipos B).

3.1 Reglas de Tipo A:

Se presentan en la tabla 2.

Tabla 2. Reglas de tipo A.

UN-Lencep	*Entidad–relación*	*SQL*
A <TIENE> B		CREATE TABLE A (**B** varchar(30) default NULL,)
A <TIENE_UNICO> B		CREATE TABLE A (**B** varchar(30) default NULL, UNIQUE (B))
A <TIENE> B, X es_posible_valor_de B Y es_posible_valor_de B		CREATE TABLE A (**B** ENUM('X','Y'))
A <ES> B		CREATE TABLE **A** ( ) CREATE TABLE **B** ( )
A <R1> B		CREATE TABLE A ( ) CREATE TABLE B ( )

Fuente: elaboración propia.

3.2 Reglas de Tipo B

Estas reglas tienen como precondición elementos propios del diagrama de clases los cuales, dicho sea de paso, se obtienen automáticamente con otras reglas que definen Zapata *et al.* [7]. Las reglas de tipo B se presentan en la tabla 3.

4 CASO DE ESTUDIO

Con el fin de ejemplificar las reglas definidas en este artículo, se presenta un caso de estudio relacionado con el sistema de notas en un colegio. El siguiente es un relato en UN-Lencep que representa el dominio del problema.

- persona tiene_único identificación
- persona tiene nombre
- profesor es persona
- profesor tiene salario
- profesor califica examen
- profesor diseña examen
- estudiante es persona
- estudiante tiene estado
- activo es_posible_valor de estado
- retirado es_posible_valor de estado
- estudiante presenta solución
- solucion tiene_único código
- solucion tiene nota
- solucion tiene estudiante
- solucion tiene respuesta
- respuesta tiene_único código
- respuesta tiene calificación
- respuesta tiene descripción
- examen tiene tema
- examen tiene nota_máxima
- examen tiene pregunta
- examen tiene solución
- pregunta tiene porcentaje
- pregunta tiene_único número
- pregunta tiene respuesta_correcta
- pregunta tiene enunciado
- pregunta tiene_único respuesta

En la tabla 4 se muestra la forma de aplicar cada una de las reglas definidas en la sección 4. La figura 1 muestra el diagrama entidad–relación completo que se obtiene a partir del caso de estudio.

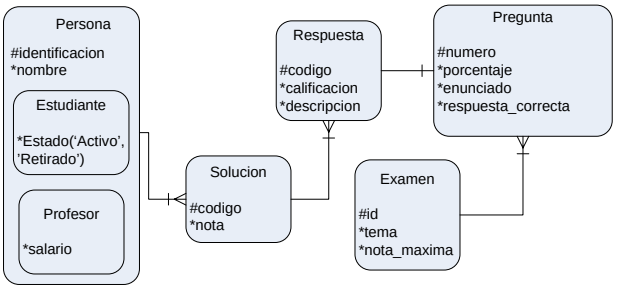
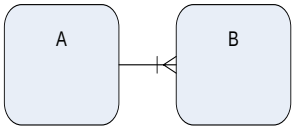
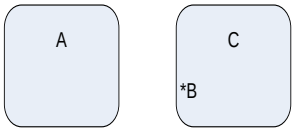


Figura 1. Diagrama entidad-relación de un sistema de notas en un colegio.

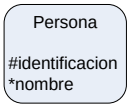
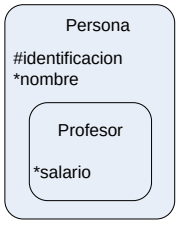
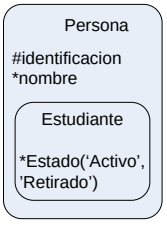
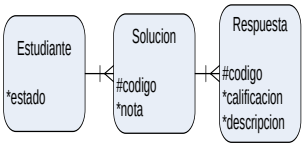
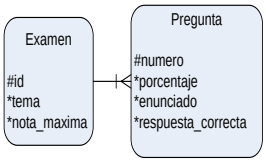
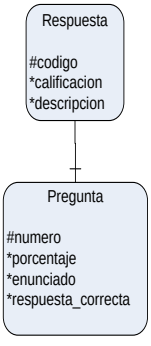
Fuente: elaboración propia.

Tabla 3. Reglas de tipo B.

UN-Lencep	*Precondición*	*Entidad–relación*	*SQL*
A <TIENE> B			CREATE TABLE **A** ( ) CREATE TABLE **B** (**a_id** int(11) FOREIGN KEY (**A**))
A <R1> B			CREATE TABLE **A** ( ) CREATE TABLE **C** (**B** varchar(30) default NULL)

Fuente: elaboración propia.

Tabla 4. Generación automática del diagrama entidad-relación y su representación en SQL a partir de un caso de estudio.

UN-Lencep	*Entidad – relación*	*SQL*
persona tiene nombre persona tiene_único identificación		CREATE TABLE Persona (identificación int(30) NOT NULL PRIMRY KEY, nombres(30) default NULL,)
profesor es persona profesor tiene salario		CREATE TABLE Profesor (salario varchar(30) default NULL, persona_id int(30) FOREIGN KEY (Persona),)
estudiante es persona estudiante tiene estado activo es_un_posible_valor de estado retirado es_un_posible_valor de estado		CREATE TABLE Estudiante(estado ENUM(‹activo›,›retirado›), persona_id int(30) FOREIGN KEY (Persona),)
solución tiene_único código solución tiene nota solución tiene respuesta		CREATE TABLE Solucion (codigo int(30) NOT NULL PRIMRY KEY, nota varchar(30) default NULL, estudiante_id int(30) FOREIGN KEY (Estudiante),)
respuesta tiene_único código respuesta tiene calificación respuesta tiene descripción		
examen tiene tema examen tiene nota_máxima examen tiene_único id examen tiene preguntas examen tiene solución		CREATE TABLE Examen (id int(30) NOT NULL PRIMRY KEY, nota_máxima varchar(30) default NULL,)
pregunta tiene porcentaje pregunta tiene_único número pregunta tiene respuesta_correcta pregunta tiene enunciado pregunta tiene_único respuesta		CREATE TABLE Pregunta (número int(30) NOT NULL PRIMRY KEY, porcentaje varchar(30) default NULL, enunciado varchar(30) default NULL, respuesta_correcta varchar(30) default NULL, respuesta_id int(30) FOREIGN KEY (Respuesta), examen_id int(30) FOREIGN KEY (Examen),)

Fuente: elaboración propia.

Al ser un proceso automático, el analista no debe preocuparse por la aplicación de las reglas de conversión. De esta manera, se evitan errores humanos que se puedan cometer. Nótese que, a excepción de las relaciones dinámicas (acciones), cada especificación en UN-Lencep se refleja en el diagrama entidad-relación y, por lo tanto, en el SQL correspondiente.

5 CONCLUSIONES Y TRABAJO FUTURO

En este artículo, se presentó un conjunto de reglas heurísticas que permiten obtener automáticamente el diagrama entidad-relación y su representación en SQL, a partir de un lenguaje natural controlado (UN-Lencep). Algunos de los principales aportes de este trabajo son:

- Al ser un proceso automático, se evitan errores humanos en la aplicación de las reglas de conversión.
- Se reducen tiempo y costos en el desarrollo de una aplicación.
- Se mejora la comunicación con el interesado, dado que el lenguaje natural controlado (UN-Lencep) es de fácil comprensión y no requiere previa capacitación.
- El interesado puede proveer una validación de la información que se representa en UN-Lencep en las etapas iniciales del proceso de desarrollo de software.

Las líneas de trabajo futuro que se pueden derivar de este trabajo son:

- Desarrollar un prototipo, que implemente las reglas definidas en este artículo, además de generarlos diferentes diagramas UML que establecen Zapata *et al.* [7].
- Definir nuevas reglas heurísticas con el fin de obtener SQL para varios gestores de bases de datos (ORACLE, MYSQL, PostgreSQL).
- Definir reglas para la generación de *triggers* con el fin de validar procesos no solo en la aplicación sino también en la base de datos.
- Definir reglas para la definición de tipos de datos (*varchar, int, boolean*, etc.).

6 AGRADECIMIENTOS

Este trabajo se financió parcialmente con fondos de la Vicerrectoría de Investigación de la Universidad Nacional de Colombia, mediante el proyecto de investigación "Transformación semiautomática de los esquemas conceptuales, generados en UNC-Diagramador, en prototipos funcionales".

REFERENCIAS

[1] P. Chen, "English sentence structure and entity relationship diagrams," *Information Sciences*, vol. 29, no. 2, pp. 127-149, 1983.

[2] E. Buchholz, y A. Düsterhöft, "Using natural language for database design," presentado a Deutsche Jahrestagung für Künstliche Intelligenz, Bonn, 1994, pp. 5.

[3] N. Omar *et al.*, "Heuristics-based entity-relationship modelling through natural Language processing," en Fifteenth Irish Conference on Artificial Intelligence and Cognitive Science (AICS-04), Galway, Ireland, 2004.

[4] F. Gomez *et al.*, "A system for the semiautomatic generation of ER models from natural language specifications," *Data and Knowledge Engineering*, vol. 29, no. 1, pp. 57-81, 1999.

[5] E. Christoph, y P. C. Lockemann, "Acquisition of Terminology Knowledge Using Database Design Techniques," en ACM SIGMOD Conference, New York, 1985.

[6] A. Gangopadhyay, "Conceptual modeling from natural language functional specifications," *Artificial Intelligence in Engineering*, vol. 15, no. 2, pp. 207-218, 2001.

[7] C. M. Zapata *et al.*, "UN–Lencep: Obtención automática de diagramas UML a partir de un lenguaje controlado," presentado a 3er Taller en tecnologías del Lenguaje Humano del Encuentro Nacional de Computación, San Luis Potosí, 2006.

[8] F. Bodart *et al.*, "Architecture Elements for Highly-Interactive Business-Oriented Applications," en Lecture Notes in Computer Science, Moscú, 1993.

[9] F. Bodart, y J. Vanderdonckt, "On the Problem of Selecting Interaction Objects." in a Proceedings of BCS Conf. HCI'94 People and Computers IX, Cambridge: Cambridge University Press, pp. 163-178, 1994.

[10] F. Bodart *et al.*, "Towards a Systematic Building of Software Architectures: the TRIDENT methodological guide." in 2nd Eurographics Workshop on Design, Specification, Verification of Interactive Systems DSV-IS'95, Viena: Springer- Verlag, pp. 262-278, 1995.

[11] J. Vanderdonckt, y F. Bodart, "Encapsulating Knowledge for Intelligent Automatic Interaction Objects Selection." in Proceedings of the Conference on Human Factors in Computing Systems InterCHI'93 "Bridges Between Worlds", Amsterdam: ACM Press, pp. 424-429, 1993.