Metodología para el diseño de una base de datos de modelo CAD basado en STEP.
Methodology for the design of a CAD model database based in STEP.

Ing. Erodis Pérez Michel
Profesor Instructor
Universidad de Granma, Departamento de Informática, Bayamo, Granma, Cuba,
Telf: (0123) 48 81 41 Email: eperezm@udg.co.cu

Dr.C. Ricardo Lorenzo Ávila Rondón
Profesor Titular
Universidad de Holguín, Centro de Estudios CAD/CAM, Holguín, Cuba.
Telf: (0124) 48 26 78 Email: ricardo@cadcam.uho.edu.cu

Recibido: 09-09-14
Aceptado: 27-10-14

Resumen:

Este artículo presenta una metodología o estrategia para el modelado de una base de datos de modelo CAD basado en STEP. Este modelo esta descrito en el lenguaje de modelado EXPRESS. Se utiliza el Protocolo de Aplicación 203 como modelo CAD y a la herramienta libre PostgreSQL, aprovechándose las facilidades de su modelo de datos objeto-relacional. También se considera las técnicas y los patrones de mapeo objeto-relacional y otros conceptos de análisis y diseño orientado a objeto.

Palabras clave: Base de datos, STEP, EXPRESS, PostgreSQL, Mapeo objeto-relacional

Abstract:

This paper presents a methodology or approach for modeling a CAD model database based on STEP. EXPRESS language describes the model. It is utilized the Application Protocol 203 like CAD model and the free tool PostgreSQL, taking advantage of facilities of its own object-relational data model. Techniques and patterns of object-relational mapping and other concepts of object oriented analysis and design are considered.

Keywords: Database, STEP, EXPRESS, PostgreSQL, Object-relational mapping

Introducción:

Con la introducción de las tecnologías de la información en las empresas, una de las áreas más influenciadas es la del diseño mediante la utilización de las herramientas de Diseño Asistido por Computadora (CAD, de **Computer Aided Design**), a través de los cuales se ha logrado mejorar significativamente la calidad y rapidez de los diseños.

Los dibujos creados con herramientas CAD representan aumentos de productividad sobre los dibujos en papel, así como la facilidad de revisarlos y archivarlos. Las herramientas CAD también abrieron nuevas oportunidades, como la habilitación de instrucciones de fabricación que se derivan de forma automática y se ejecuta directamente desde el dibujo a través de los sistemas de Planeación de la Producción Asistida por Computadora (CAPP, de **Computer Aided Production Planning***)*.

Las herramientas CAD, CAPP y CAM (del inglés **Computer Aided Manufacturing**) se han proliferado para satisfacer necesidades de ingeniería cada vez más complejas y diversas, y con estas los formatos que utilizan para almacenar e intercambiar los datos del producto.

Para que las empresas compartan el diseño de un producto a través de diversos sistemas CAD, CAPP, CAM y otros que abarcan el ciclo de vida de este, lo ideal sería que existiera un formato estándar para transferir la información de manera que las herramientas puedan reconocer los datos del producto. Este requisito resulta cada vez más indispensable en una era donde los fabricantes suelen formar empresas conjuntas para hacer frente a oportunidades de negocio, y donde los socios de una cadena de suministros están llamados a ofrecer una gama cada vez más compleja de servicios.

Existen un conjunto de estándares para intercambiar información entre diferentes sistemas que intervienen durante el ciclo de vida de un producto, tal es el caso de los ficheros de intercambio DXF y DWG de la compañía Autodesk o los estándares internacionales IGES (del inglés Initial Graphics Exchange Specification) [1] o STEP (acrónimo de Standard for the Exchange of Product model data) [2]; sin embargo los estándares de intercambio IGES o STEP han sido los que más han logrado imponerse como método para intercambiar información de forma de productos. El objetivo de estos estándares es garantizar la interoperabilidad entre los diferentes sistemas CAD/CAPP/CAM. La interoperabilidad persigue facilitar, compartir e intercambiar información del producto entre varios módulos dentro de un sistema de desarrollo del producto [3].

La norma STEP es una de las mejores estructurada y documenta, lo que la hace adecuada no sólo para el *intercambio* de archivos neutro con información del producto, sino también como base para implementar y *compartir* bases de datos de productos. La integración de los datos del producto entorno a una base de datos puede propiciar la ingeniería concurrente, un proceso donde múltiples ingenieros trabajan en varias facetas del producto concurrentemente [4], no obstante las bases de datos integradas del producto no son comunes en la industria. La razón de esto es que las aplicaciones de ingeniería manejan usualmente modelos de información muy complejos. Estos modelos son complejos porque las aplicaciones de ingeniería manipulan simulaciones del mundo real y estos modelos poseen información referente a la geometría, la tolerancia, los materiales y los planes de manufactura, etc. que son estructuralmente y semánticamente muy ricos.

Ahora bien se han desarrollado varias investigaciones para compartir datos de modelos CAD del producto entorno a una base de datos; pero no existe una herramienta libre que permita compartir en una base de datos común, modelos de información CAD descritos en EXPRESS entre sistemas CAD/CAPP y otros.

Métodos y Materiales

La revisión de fuentes bibliográficas, esencialmente de la norma ISO 10303, de conjunto con el método analítico – sintético permitieron sintetizar los elementos más importantes relacionados con el diseño de base de datos de modelos CAD basados en la norma STEP [5], los elementos esenciales del modelo de datos objeto - relacional del sistema de base de datos PostgreSQL y las técnicas o patrones de diseño para el mapeo objeto/relacional.

ISO 10303 o STEP (Estándar para el intercambio de modelo de datos de productos)

Es un estándar internacional para la representación e intercambio de información de productos industriales. El objetivo es proveer un mecanismo que sea capaz de describir la información de un producto a través de su ciclo de vida, independientemente de cualquier sistema en particular. La naturaleza de esta descripción la convierte en la adecuada para intercambiar y compartir modelos de datos de productos. Uno de los aspectos más importantes de STEP es su extensibilidad. Esta extensibilidad es el resultado de la decisión de basar a STEP en un lenguaje de modelado de información, el lenguaje EXPRESS [6], el cual constituye el método de descripción fundamental de la norma y es puramente orientado a objeto. El estándar STEP clasifica los tipos diversos de datos del producto en Protocolos de Aplicación (AP). Cada AP es un documento formal que describe las actividades en el ciclo de vida de un producto. Uno de estos protocolos es el Protocolo de Aplicación 203 "Configuration Controlled 3D Designs of Mechanical Parts and Assemblies" [7], el cual se refiere a la transferencia de información de modelos de forma del producto, estructura de ensamble y control de configuración, por ejemplo: versiones de pieza, estado de liberación, los datos de autorización.

Mapeo Objeto/Relacional

El mapeo objeto-relacional, más conocido por su nombre en inglés Object-Relational mapping o sus siglas O/RM, ORM y O/R mapping, es una técnica de programación para convertir datos entre el sistema de tipos utilizado en un lenguaje de programación orientado a objetos, y el utilizado en una base de datos relacional, utilizando un motor de persistencia [8]. En la práctica esto crea una base de datos orientada a objetos virtual sobre la base de datos relacional. Esto posibilita el uso de las características propias de la orientación a objeto, básicamente la herencia y el polimorfismo.

Debido a que EXPRESS es un lenguaje para el modelado que hace uso de los conceptos de orientación a objeto antes mencionados para describir los datos de un producto determinado, es preciso para realizar un correcto mapeo de modelos CAD descritos en este lenguaje, tener en cuenta los patrones y técnicas de mapeo objeto-relacional.

Para una discusión detallada sobre los beneficios e inconvenientes de los patrones y técnicas de mapeo objeto-relacional véase [9, 10]. Para obtener más información acerca de la implementación de estos modelos consúltese [11].

El uso de un ORM es una alternativa sumamente efectiva a la hora de trasladar el modelo orientado a objeto al esquema relacional nativo de las bases de datos SQL. Evita la inclusión de sentencias SQL embebidas en el código de la aplicación, lo que a su vez facilita la migración hacia otro sistema gestor de bases de datos. Incorpora además una capa de abstracción entre el modelo relacional físico y la capa de negocios de la aplicación. Al ser realizado en esta capa de manera automática la conversión de instrucciones orientadas a objetos a sentencias SQL, se minimiza la ocurrencia de errores humanos.

Sistema de base de datos PostgreSQL

Es un potente sistema de gestión de bases de datos objeto-relacional (O-RDBMS), multiusuario, centralizado y de propósito general, que está siendo desarrollado desde 1977 y está liberado bajo la licencia Berkeley Software Distribution (BSD). Es ampliamente considerado como el sistema gestor de bases de datos de código abierto más avanzado del mundo. Ofrece sofisticadas características muy útiles a la hora de concebir una herramienta ORM que sirva como capa intermedia entre el modelo orientado a objeto y el esquema relacional nativo de las bases de datos SQL, algunas de las características se mencionan a continuación: organiza los datos mediante un modelo objeto-relacional; Soporte para parte de los estándares SQL 92, 99, 2003 y 2008. Incorpora comportamiento para manejar colecciones de valores en los campo de una tabla o lo que se conoce como tablas Non-First Normal Form [12]. Cuenta con una API sumamente flexible

para el trabajo con varios lenguajes de programación procedurales como: C, C++, NET, Bash, Delphi, PL/Java, PL/Perl, PL/Tcl, PL/pgSQL, PL/Ruby, PL/PHP, PL/Python, PL/Scheme y PL/R. Ofrece transacciones que permiten el paso entre dos estados consistentes manteniendo la integridad de los datos. Tiene incorporado el mecanismo Write Ahead Logging (WAL), que incrementa la confiabilidad de las bases de datos al registrar los cambios antes de ser escritos al disco; lo que asegura que, en caso de ocurrir un fallo crítico en las bases de datos, exista un registro de transacciones del cual se pueda restaurar [13].

Resultados

Como resultado de esta investigación se obtiene una metodología o estrategia para modelar bases de datos de modelos CAD descritos en el lenguaje EXPRESS, usando herramientas libres y que aprovecha las facilidades del sistema de base de datos objeto-relacional PostgreSQL, las técnicas de mapeo objeto-relacional y otros conceptos de análisis y diseño orientado a objeto.

Estrategia para definir la estructura de la base de datos relacional desde EXPRESS

Los modelos de información EXPRESS orientados a objeto y el modelo relacional son paradigmas diferentes de programación. Cuando los objetos necesitan ser almacenados en bases de datos relacionales, las diferencias entre estas dos perspectivas necesitan ser minimizadas. Si sólo los módulos de abstracción de datos necesitan ser mapeados a una base de datos relacional el proceso es relativamente fácil. Pero cuando existen modelos de objeto completamente solapados los conceptos de programación orientada a objeto tienen que ser mapeados a la estructura relacional de tabla. Estos conceptos son:

- la herencia y el polimorfismo,
- las interrelaciones entre las clases (asociación, agregación, o composición),
- y los tipos de datos más sofisticados que los tipos de datos SQL.
- Otros, pueden consultarse en [14, 15].

Cada uno de los conceptos antes mencionados pueden ser mapeados usando soluciones diferentes para el mismo problema. Cada solución diferente será tratada como un patrón separado, véase [9, 10].

Para definir la estructura de la base de datos relacional, los implementadores deben convertir un modelo de información EXPRESS en las definiciones del esquema para la base de datos de destino. Esta conversión requiere un mapeo de las construcciones semánticas del lenguaje EXPRESS en el modelo de datos del sistema de bases de datos de destino. Los modelos de información EXPRESS pueden retar las capacidades de los sistemas de bases de datos existentes. En particular, las siguientes características de EXPRESS pueden requerir codificación u otras manipulaciones para preservar la información original en el modelo de datos nativo:

- **Entidades (ENTITY en EXPRESS)**: Existen varias alternativas para transformar las entidades y su jerarquía de herencia, una de ellas es transformar cada entidad a una tabla en la base de datos relacional, para esto es preciso agregar un atributo que identifique a cada tupla en la tabla, es decir una llave primaria. También hay que tener en cuenta que una entidad puede tener atributos simple y otros más complejos como agregaciones de otras entidades o atributos que son colecciones de valores, los cuales no son soportados directamente por los Sistema de Gestión de Base de Datos actuales. Los atributos de una entidad son representados como una columna en la tabla de la entidad o como una tabla. Si el atributo es una asociación (los atributos cuyo tipo es **array, bag, list** o **set**) tiene su propia tabla; de otra manera, el atributo es representado como una columna de la tabla entidad. La plantilla de la tabla 1 resume la estructura de la tabla entidad, cuyo nombre será el nombre de la entidad en el lenguaje EXPRESS.

Tabla 1. Estructura de la tabla entidad.

Atributo del sistema	**Atributos para las entidades base**		**Atributos simples**				
EntityID	**Model_ID**	**EntityRef**	**Columnas de atributos propios**				

Descripción de los atributos columnas

La primera columna de cada tabla entidad es **EntityID**. Contiene un identificador único para cada instancia de una entidad. Este identificador es utilizado como la llave primaria de la tabla entidad y este probablemente pueda ser referenciado en dos situaciones fuera de esta tabla. Una entidad referenciada por otra entidad como un atributo representado por este identificador en la columna de ese atributo de la tabla entidad. Para una entidad con atributos agregado, el mismo identificador de la entidad es también usado en las tablas que contienen los datos para estos atributos.

La segunda columna de cada tabla entidad que es base en el árbol de herencia es **Model_ID**, un atributo llave foránea que se agrega para identificar a que producto pertenece cada instancia de una entidad.

La tercera columna de cada tabla entidad que es base en el árbol de herencia es **EntityRef.** Este atributo contiene la referencia a una instancia de una entidad según lo establecido por el estándar **Parts 21** [16]**.**

Finalmente, las restantes columna serán solo para los tipos básicos de EXPRESS que puedan ser transformados directamente a tipos de datos en SQL del sistema de base de datos relacional destino, o sea, los atributos no agregado declarados directamente en la definición de la entidad EXPRESS son columnas de la tabla.

En el ejemplo que sigue a la porción del esquema EXPRESS son mostradas las declaraciones SQL producidas para crear las tablas necesarias para el esquema EXPRESS. La tabla 2 resume los tipos básicos EXPRESS y su posible representación en el sistema de base de datos objeto-relacional de código abierto PostgreSQL [17].

Ejemplo en EXPRESS**:**

ENTITY geometry (* GEOM-1 *)
SUPERTYPE OF (**ONEOF** (point, vector, curve, surface,

coordinate_system, transformation, axis_placement));

local_coordinate_system: **OPTIONAL** coordinate_system;

axis: **OPTIONAL** transformation;

END_ENTITY;

ENTITY vector (* GEOM-3 *)
SUPERTYPE OF (direction **ANDOR** vector_with_magnitude)

SUBTYPE OF (geometry);

END_ENTITY;

```
ENTITY direction (* GEOM-14 *)
        SUBTYPE OF (vector);

        x: REAL;

        y: REAL;

        z: OPTIONAL REAL;

END_ENTITY;
```

Ejemplo en SQL**:**

```
CREATE TABLE geometry (
        EntityID BIGSERIAL PRIMARY KEY /*SYSTEM ATTRIBUTES*/,

        Model_ID BIGSERIAL NOT NULL /*SYSTEM ATTRIBUTES*/,

        EntityRef VARCHAR (50) NOT NULL /*SYSTEM ATTRIBUTES*/,

        local_coordinate_system BIGSERIAL /* FOREIGN KEY */,

        axis BIGSERIAL /* FOREIGN KEY */
);

ALTER TABLE geometry ADD CONSTRAINT  FK_ geometry_LCS
        FOREIGN KEY (local_coordinate_system)
        REFERENCES coordinate_system (EntityID);

ALTER TABLE geometry ADD CONSTRAINT FK_ geometry_Transf
        FOREIGN KEY (axis)
        REFERENCES transformation (EntityID);
CREATE TABLE vector (
        EntityID BIGSERIAL PRIMARY KEY /* FOREIGN KEY */
);

ALTER TABLE vector ADD CONSTRAINT FK_ vector_ID
        FOREIGN KEY (EntityID)
        REFERENCES geometry (EntityID);


CREATE TABLE direction (
        EntityID BIGSERIAL PRIMARY KEY /* FOREIGN KEY */,

        X FLOAT8 NOT NULL,

        Y FLOAT8 NOT NULL,

        Z FLOAT8
);

ALTER TABLE direction ADD CONSTRAINT FK_ direction_ID
        FOREIGN KEY (EntityID)
        REFERENCES vector (EntityID);
```

Tabla 2. Coincidencia de tipos base EXPRESS y SQL en PostgreSQL.

EXPRESS	PostgreSQL
INTEGER	INTEGER
REAL	FLOAT8, DOUBLE PRECISION
REAL(n)	NUMERIC [(n, s)], DECIMAL [(n, s)]
NUMBER	FLOAT8, DOUBLE PRECISION
BOOLEAN	CHAR (3), SMALLINT, BOOL
LOGICAL	CHAR (3), SMALLINT
STRING	TEXT (alrededor de 1 Gb)
STRING (n)	VARCHAR (n)
STRING (n) FIXED	CHAR (n)
BINARY	BYTEA
BINARY (n)	VARBIT (n)
BINARY (n) FIXED	BIT (n)
ENTITY (referencia)	VARCHAR (50), FOREIGN KEY

Descripción de los atributos tablas

Los atributos de una entidad que son representados como una tabla, son aquellos que representan una asociación con otra entidad, o sea los atributos cuyo tipo es **array, bag, list** o **set**.

PostgreSQL permite que columnas de una tabla sean definidas como matrices multidimensionales de longitud variable [18]. En este caso aprovechando esta característica del modelo de datos objeto - relacional de PostgreSQL, para las columnas que son asociaciones con tipos bases de EXPRESS como se definió en la Tabla 2, el proceso de traducción para esos atributos es similar a los atributos con tipo de datos básicos, solo hay que definir esos atributos como un arreglo del tipo básico en PostgreSQL, sin importar si son de tipo **array, bag, list** o **set**. Sin embargo hay que tener en cuenta que los tipos **set** y **bag** no permiten valores nulos. Un ejemplo de este proceso se muestra en la tabla 3 que aparece a continuación:

Tabla 3. Reglas para la traducción de las asociaciones de tipos base.

EXPRESS	PostgreSQL
ENTITY point coord: **ARRAY** [3] **OF REAL**; **END_ENTITY;**	**CREATE TABLE** point (EntityID **BIGSERIAL PRIMARY KEY**, Model_ID **BIGSERIAL NOT NULL**, EntityRef **VARCHAR** (50) **NOT NULL**, coord **FLOAT8** [3]); **Nota:** • El atributo **coord** almacenará el valor de la coordenada **x, y, z** del punto en cuestión.

Para el caso de las asociaciones más complejas es preciso tener en cuenta las siguientes consideraciones:

- El nombre de la tabla es una combinación del nombre de la entidad y el nombre del atributo, de esta manera se asegura que el nombre de la tabla sea único.
- Cada tupla en la tabla tendrá una llave foránea, ***fk_aggr_column,*** el cual hace referencia a la llave primaria de la instancia en la entidad que tiene la agregación del atributo.
- El campo ***index_column*** tiene un valor ordinal de los elementos de las colecciones ordenadas tales como las listas y los arreglos.
- La agregación anidada tendrá varios campos ***index_column*** en dependencia del nivel de anidación.
- El campo ***value_column*** almacenará la llave foránea de la tabla en que se almacena el valor realmente.

Un resumen de las consideraciones anteriores se muestra en la siguiente tabla:

Tabla 4. Estructura de la tabla para las asociaciones entre entidades.

Nombre de la tabla:		**TableName_AttributeName**	
id_column (“ID”)	**fk_aggr_column** (*“fk_owner”*)	**index_column** (“attribute_number”)	**value_column** ("attribute_name")

- **Los tipos de datos construidos:** hay dos clases de tipos de datos construidos en EXPRESS: el tipo de dato **ENUMERATION** y **SELECT**. Estos tipos de datos tienen estructuras sintácticas similares y se usan para proveer representaciones subyacentes de tipos de datos definidos por el usuario. Ambos indican la especificación de un dominio para atributos. Un tipo ENUMERATION especifica los valores posibles para el dominio explícitamente; un tipo SELECT especifica los valores posibles indirectamente.

Procedimiento para la traducción de los ENUMERATION a ENUM en PostgreSQL

El tipo de datos **ENUMERATION** tiene como su dominio un conjunto de nombres. Los nombres representan valores del tipo de datos. Estos nombres son nombrados **enumeration_ids** y son llamados ENUMERATION ítems.

Para traducir un tipo ENUMERATION el sistema gestor de base de datos PostgreSQL posee una instrucción [19] para definir un tipo similar a este, y que después puede ser usado en la definición de otros objetos en un esquema de base de datos. Por ejemplo si se tiene la declaración en EXPRESS siguiente:

```
TYPE b_spline_curve_form = ENUMERATION OF
                (polyline_form, circular_arc, elliptic_arc,
                 parabolic_arc, hyperbolic_arc, unspecified);
END_TYPE;
```

para definir el tipo **b_spline_curve_form** el cual puede ser usado en la definición de otras entidades. Para traducirlo a un tipo equivalente en SQL, se utiliza la instrucción **CREATE TYPE** y se hace uso del tipo básico **ENUM**, con el cual se define un nuevo dominio y los valores correspondientes para un atributo determinado. La instrucción DDL correspondiente se muestra a continuación:

```
CREATE TYPE b_spline_curve_form AS
```

ENUM ('polyline_form', 'circular_arc', 'elliptic_arc', 'parabolic_arc', 'hyperbolic_arc', 'unspecified');

De esta forma una vez ejecutada la instrucción quedaría listo para usarse el nuevo tipo.

Procedimiento para la traducción del tipo SELECT

El tipo de datos **SELECT** tiene como su dominio la unión de los dominios de los tipos de datos designados en su lista de selección. Este es una generalización de cada uno de los tipos de datos designados en su lista de selección. A continuación se muestra un ejemplo en EXPRESS de la descripción del tipo SELECT **geometric_set_select**, cuyo valor está definido por el valor de los tipos en su lista de selección, **point, curve, surface**.

TYPE geometric_set_select = **SELECT**

(point, curve, surface);

END_TYPE;

En PostgreSQL no existe un tipo equivalente, por tanto cuando los tipos de la lista de selección son entidades, los valores del campo en la tabla son los identificadores de entidad (**EntityID**) como si el tipo del atributo hubiese sido una entidad. Por otra parte, cuando los tipos de la lista de selección no son entidades, los valores deben ser del mismo tipo base que el tipo definido. En EXPRESS es posible crear un objeto que no tiene un solo tipo base a través del uso del tipo SELECT. En ésta propuesta de diseño no se tiene en cuenta esto porque en la especificación del Protocolo de Aplicación 203 las definiciones de la mayoría de los tipos de datos SELECT, la selección es entre tipos entidad; por consiguiente, un atributo de tipo SELECT se traduce en el tipo SQL BIGSERIAL, igual que los atributos identificadores del tipo entidad. Finalmente en la tabla que contiene atributos que son de tipo SELECT se le agregará un atributo (**SL_attribute**) por cada uno de los atributos de tipo SELECT, estos atributos almacenarán la descripción del tipo seleccionado en la lista de selección del tipo SELECT. La tabla 5 especifica la regla de traducción descrita anteriormente.

Tabla 5. Reglas para la traducción del tipo SELECT.

EXPRESS [20]	**PostgreSQL**
ENTITY geometric_set **SUBTYPE OF** (geometric_representation_item); element: *geometric_set_select*; **END_ENTITY;**	**CREATE TABLE** geometric_set (EntityID **BIGSERIAL PRIMARY KEY**, element **BIGSERIAL** /* SELECT type*/, SL_element **VARCHAR** (128)); **Nota:** • El atributo **element** almacenará el valor de la llave de la entidad tabla con que fue instanciado (**point, curve, surface**). • **SL_element** almacenará precisamente el nombre del valor seleccionado entre (**point, curve, surface**).

Discusión

No existe un método de implementación estándar para traducir modelos STEP en un modelo de datos relacional, sin embargo tal traducción es factible y se pueden encontrar algunos caso que así lo demuestran. Los esfuerzos desarrollados en este sentido han sido encaminados fundamentalmente a la implementación de bases de datos basadas en la norma ISO 10303, aprovechando el método de descripción de la norma, el lenguaje EXPRESS. Numerosas investigaciones se han realizado en este sentido en diferentes etapas dentro de las que destacan las siguientes:

JSDAI Database es una solución proporcionada por LKSoft [21] para almacenar información sobre los productos industriales como objetos en bases de datos relacionales, y hacerla disponible a través de interfaces de programación de aplicaciones (API) compatibles con los estándares. Cada tipo de objeto de producto especificado por un modelo de información EXPRESS es compatible. Esto incluye las normas internacionales conocidas como STEP o ISO 10303, la biblioteca de piezas (ISO 13584 o PLIB), los tipos de elementos de datos estándar (IEC 61360) y otros. Además admiten esquemas EXPRESS específicos de un usuario. Es software propietario que utiliza la estrategia de transformación de tipos de datos del modelo a tabla en la base de datos, cuyo inconveniente es que los datos de una entidad están disperso por varias tablas en la base de datos, incrementado la complejidad de la extracción de la información de la base de datos.

Babu y otros en este artículo [22] proponen una bases de datos de manufactura para datos STEP-NC [23] de entidades EXPRESS. Esta base de datos de manufactura principalmente incluye datos de procesamiento, datos de manufactura para el maquinado y el torneado y los datos del herramental para estas operaciones tecnológicas. En este artículo no se describe el procedimiento usado por los autores para transformar el esquema EXPRESS de STEP-NC al lenguaje de base de datos de destino.

You y otros presentan la implementación de GTCIS2SQL [24] una base de datos relacional sobre CIS/2, un estándar de modelado de producto para estructuras de acero usadas en la construcción de edificios. GTCIS2SQL soporta comunicación basada en la Web con aplicaciones que intercambian datos en archivos neutros **p21**. Este soporta una gran variedad de facilidades para consultar los datos. Este artículo describe el procedimiento y las consideraciones hechas a las construcciones del lenguaje EXPRESS para construir el esquema de la base de datos que almacenará los datos de un producto descrito con este estándar.

Loffredo propone [25] una guía para implementar bases de datos de ingeniería alrededor de modelos complejos. En particular, este trabajo presenta una evaluación sistemática de las arquitecturas de implementación de STEP, un set de estándares de comparación que simulan como las aplicaciones de ingeniería acceden a una base de datos de modelos STEP y las recomendaciones extraídas de los experimentos con varios sistemas de bases de datos excepto con PostgreSQL.

Morris en su reporte de trabajo describe el método usado por el software **fedex_sql** [26] para traducir un esquema EXPRESS en las declaraciones SQL, las cuales generen un esquema de base de datos relacional para almacenar datos STEP. El software fue desarrollado por el NIST (National Institute Of Standards And Technology). También presenta tres tipos de dificultades que están involucradas en la traducción de esquemas EXPRESS a un esquema de base de datos relacional: la traducción de las construcciones semánticas de EXPRESS en el lenguaje de definición de datos de SQL, la resolución de limitación impuesta por el sistema de gestión de base de datos, y el desarrollo de un diccionario de datos. En el artículo se le da solución a las dos primeras limitaciones. Sin embargo el proyecto fue abandonado y no existe soporte para el mismo.

Existen otras investigaciones al respecto [27-30] que fundamentalmente basan su trabajo en la traducción de esquemas EXPRESS a sistemas de bases de datos orientados a objetos, siendo esto una vía factible si se tiene en cuenta que los modelos de datos de los protocolos de aplicación poseen ciento y miles de definiciones de entidades y tipos de datos redefinidos, pero su único inconveniente es que los sistemas de base de datos orientados a objetos no son muy comunes y la mayoría de los existentes son software propietario.

La metodología propuesta en esta investigación aprovecha cada una de las ventajas e inconveniente de estas investigaciones, además saca provecho del modelo de datos híbrido objeto-relacional del sistema de base de datos PostgreSQL para representar algunas construcciones semánticas del lenguaje de modelado EXPRESS y se basa en el uso de herramienta en la modalidad de software libre.

Conclusiones:

La metodología propuesta constituye una herramienta para desarrollar un ORM que sirva como capa intermedia entre el modelo orientado a objeto obtenido a partir del modelo EXPRESS de un protocolo de aplicación y el modelo relacional que implementa el sistema de base de datos PostgreSQL. También constituye un método robusto de mapeo de las construcciones semánticas del lenguaje de modelado EXPRESS y la implementación del lenguaje SQL del sistema gestor de base de datos PostgreSQL. Además es una alternativa muy efectiva para simplificar la persistencia de la información y lograr la abstracción del sistema gestor de base de datos. Reduce el número de tablas en la base de datos al utilizar el tipo ARRAY de PostgreSQL.

Referencias:

1. Kamrani, A. Y Nasr, E.A., *IGES Standard Protocol for Feature Recognition CAD System*, en *Rapid Prototyping: Theory and Practice*. 2006, Springer. p. 323.
2. ISO10303, *Industrial automation systems and integration – Product data representation and exchange* 1994.
3. Szykman, S., et al., *A foundation for interoperability in next-generation product development systems.* Computer Aided Design, 2001. **33**(7): p. 545–559.
4. Schenck, D. Y Wilson, P., *Information Modeling the EXPRESS Way*. 1994: Oxford University Press.
5. ISO10303-1, *Industrial automation systems and integration – Product data representation and exchange* en *Part 1: Overview and fundamental principles.* 1994.
6. ISO10303-11, *Industrial automation systems and integration – Product data representation and exchange* en *Part 11 : EXPRESS Language*. 1994.
7. ISO10303-203, *Industrial automation systems and integration – Product data representation and exchange*, en *Part 203. Aplication Protocol: Configuration Controlled 3D Designs of Mechanical Parts and Assemblies*. 1994.
8. Simitci, A. *Object-Relational Mapping in Database Design*. 2012.
9. Amber, S.W. *Agile database techniques*. 2003 [Consultado: 2014, 18 de Enero]; Disponible en: http://www.agiledata.org/essays/mappingObjects.html,Wiley.
10. Keller, W. *Mapping objects to tables: A Pattern Language*. Object Architects, 1997.
11. Fowler, M., *Patterns of Enterprise Application Architecture*. 2002: Adisson-Wesley.
12. Matthew, N. Y Stones, R., *Beginning Databases with PostgreSQL*. Second ed. From Novice to Professional, ed. J. Gilmore. 2005, New York: Apress. 661.
13. Martínez, R. *Portal sobre PostgreSQL en español*. 2010 02/10/2010 [Consultado: 2014, 29 de Mayo]; Disponible en: http://www.postgresql.org.es/sobre_postgresql.
14. Booch, G., *Object-oriented analysis and design with applications*. 2nd ed. 1998: Addison Wesley Longman, Inc. 553.
15. Abián, M.Á., *Orientación a objetos: conceptos, terminología y lenguajes(Parte 2)*. 2006, www.javahispano.org Internet.
16. ISO10303-21, *Industrial automation systems and integration – Product data representation and exchange* en *Part 21. Implementation methods: Clear text encoding of the exchange structure*. 1994.
17. PostgreSQL GDG, *Data Types*, en *PostgreSQL Documentation*, PostgreSQL Global Development Group, Editor. 2014, University of California: California.
18. PostgreSQL GDG, *Arrays*, en *PostgreSQL Documentation*, PostgreSQL Global Development Group, Editor. 2014, University of California: California.

19. PostgreSQL GDG, *Enumerated Types*, en *PostgreSQL Documentation*, PostgreSQL Global Development Group, Editor. 2014, University of California: California.
20. ISO10303-42, *Industrial automation systems and integration – Product data representation and exchange* en *Part 42: Integrated generic resources: Geometric and topological representation*. 1994.
21. LKSoft. *JSDAI Database*. 2014 [Consultado: 2014, 11 de Junio]; Disponible en: http://www.jsdai.net/database.
22. Babu, K.S., et al., *Development of a manufacturing databse system for STEP-NC data from EXPRESS entities.* International Journal of Engineering Science and Technology, 2010. **2**(11): p. 6819-6828.
23. ISO10303-238, *Industrial automation systems and integration – Product data representation and exchange*, en *Part 238: Application Protocols: Application interpreted model for computerized numerical controllers*. 2004.
24. You., S.J., Yang, D., y Eastman, C.M. *Relational DB Implementation of STEP based product model*. en *Building for the Future: The 16th CIB World Building Congress*. 2004. Rotterdam (Netherlands): in-house publishing.
25. Loffredo, D., *Efficient Database Implementation of EXPRESS Information Models*. 1998, Rensselaer Polytechnic Institute: Troy, New York.
26. Morris, K.C., *Translating Express to SQL: A User's Guide*. 1990, National Institute of Standards and Technology.
27. Eggers, J., *Implementing EXPRESS in SQL*. 1988, ISO.
28. Ma, Z.M. Y Wang, H., *STEP implementation of imperfect EXPRESS model in fuzzy object-oriented databases.* Fuzzy Sets Syst., 2006. **157**(12): p. 1597-1621.
29. Dong, Y., et al., *Active database support for STEP/EXPRESS models.* Journal of Intelligent Manufacturing, 1997. **8**(4): p. 251-261.
30. Heidelberg, *Object-Oriented Database Implementation of the Fuzzy EXPRESS Model.*, en *Fuzzy Database Modeling of Imprecise and Uncertain Engineering Information*. 2006, Springer Berlin Heidelberg. p. 179-204.