UNIVERSIDAD CENTRAL "MARTA ABREU" DE LAS VILLAS
FACULTAD DE MATEMÁTICA, FÍSICA Y COMPUTACIÓN
DEPARTAMENTO DE CIENCIA DE LA COMPUTACIÓN

REGLAS DE NEGOCIO DESDE LA PERSPECTIVA DE LOS DATOS EN BASES DE DATOS RELACIONALES

Tesis presentada en opción al grado científico de Doctor en Ciencias Técnicas

MARTHA BEATRIZ BOGGIANO CASTILLO

Santa Clara, Cuba
2014

UNIVERSIDAD CENTRAL "MARTA ABREU" DE LAS VILLAS
FACULTAD DE MATEMÁTICA, FÍSICA Y COMPUTACIÓN
DEPARTAMENTO DE CIENCIA DE LA COMPUTACIÓN

REGLAS DE NEGOCIO DESDE LA PERSPECTIVA DE LOS DATOS EN BASES DE DATOS RELACIONALES

Tesis presentada en opción al grado científico de Doctor en Ciencias Técnicas

Autor: Prof. Auxiliar, Lic. Martha Beatriz Boggiano Castillo, M.Sc.
Tutor: Prof. Titular, Lic. Ramiro Alberto Pérez Vázquez, Dr.C.
Consultante: Prof. Titular, Lic. Luisa Manuela González González, Dra.C.

Santa Clara, Cuba
2014

A Dios

A la memoria de mi padre

A la memoria de mis abuelos

A mi madre

A mi hijo

.

AGRADECIMIENTOS

- A Dios: por el don de la FE, el AMOR y sus promesas que me impulsaron en este trabajo.
- A mi tutor Dr. Ramiro Pérez, por dedicarme mucho de su valioso tiempo, ser un guía indispensable, su paciencia, amistad y ejemplo.
- A mi consultante Dra. Luisa González, por sus importantes enseñanzas, por el tiempo dedicado, sus orientaciones y su ejemplo de consagración.
- Al Dr. José Ruíz Shulclopper, por ser el primero en confiar en mí para un trabajo de doctorado.
- Al Dr. Rafael Bello, por la fuerza que imprimió para que me iniciara en este camino.
- A los estudiantes que han estado colaborando en esta investigación, especialmente mis niños Alain Pérez, el primogénito en la implementación automática de nuestras reglas, desde pregrado hasta maestría, a Ariel Calderón, por brindarse a trabajar sin descanso, entusiasta, productivo y optimista desde su tesis de licenciatura hasta hoy. A Lisandra, con su trabajo de maestría. A los licenciados Alain Pereira, por su importante trabajo, Jorge Freddy, por su dedicación, Rolando, por su tesón, así como Royne, Alexander, Daimara, Reinier e Islemy por sus esfuerzos. A Ariel Alba, Aylén y Yaisel por su interés en sumarse y trabajar por avanzar. ¡Con todos ellos ha sido posible!.
- A todos los implicados en revisiones, gracias por su tiempo.
- A Dr. Febe Ciudad, por la revisión tan minuciosa y sus convenientes consejos; al Dr. Carlos García por sus críticas y orientaciones. A los doctores Rosendo Moreno, María Matilde García, Ana María García por sus revisiones y sugerencias.
- A la Dra. Beatriz López, por ser revisora y amiga.
- A la Dra. Gladys Casas, por su amistad y apoyo, así como al Dr. Ricardo Grau y Dra. María del Carmen Chávez por sus sugerencias.
- A la Dra. Natalia Martínez, amiga y coterránea, por apoyarme con tanta disposición.
- A los doctores Gerardo y Yanet, decanos, Gheisa, jefa de departamento, por la compresión y colaboración en diferentes momentos.
- A la doctora, hermana Zenaida García, por inspirarme confianza.
- A mi profesora de literatura Mercedes Pérez por sus valiosas correcciones.

- A los compañeros: doctora María Elena, dando ánimo; doctor Abel, aconsejando e impulsando; M.Sc. Yoan Pacheco y M.Sc. Isel Moreno, colaborando en lo necesario; M.Sc. Deborah Galpert, por su disposición a ayudar; M.Sc. Emilio Viamonte, por agilizar gestiones que permitieron avanzar más rápido.
- Al M.Sc. AndrésTellería por asumir un trabajo a favor de la culminación de este empeño.
- A los compañeros de Desoft en Villa Clara, especialmente a mi amiga M.Sc. Vivian Romero y a los compañeros del XETID en La Habana, por su apoyo.
- Al Consejo de Universidades Flamencas por su apoyo financiero a través del programa IUC VLIR-UCLV.
- A mi familia:
    - mi padre, que ya no está, pero siempre confió en mí.
    - mi madre, por su ayuda en todo momento y para Todo.
    - mi hijo, por estar ahí e impulsarme a continuar.
- A mi especial amigo Fidel Raúl, por su colaboración con los artículos, por el tiempo dedicado, por sus palabras de aliento.
- A mis amigos,¡ todos!; especialmente mi madrina Esperanza, Chabela, Julita Ross y mis hermanos Daylanis, María de Lourdes y Jorge Abel, por sus buenos deseos, por inspirarme confianza y por sus oraciones.
- A Rosydé por su cariño de madre, por estar pendiente de cada paso y cada logro mío para alcanzar esta meta. A Ili y Glicet por ser hermanas y amigas siempre.
- A mis familiares, tía Martha, Carmen, Xiomara y todos los hermanos del Grupo de oración de Renovación Carismática de Placetas, entre ellos Teresita, la China Justi, Elisa, Milagros, Odalys, Petra, Félix López, (la lista sería muy larga), por animarme.
- A todos los compañeros docentes y no docentes del CEI.
- A los que con sus acciones y buenos deseos han contribuido a la terminación de este trabajo, aunque no estén sus nombres, ellos lo saben, *muchas gracias*.

SÍNTESIS

El enfoque de reglas de negocio es una temática en que las reglas se administran con cierta independencia de los sistemas de información (SI) y permite la automatización de las mismas.
Existen reglas que se representan en el propio esquema de la base de datos, no sucede así con aquellas asociadas al comportamiento de los datos y sus operaciones. La implementación de las reglas generalmente sucede a su identificación basada en categorías, desde dos perspectivas: del negocio y de los SI, expresadas en los niveles de abstracción, informal y técnico. Los motores de reglas para lograr su automatización son muy populares, pero controvertidos.
En este trabajo se proponen un conjunto de categorías de reglas desde la perspectiva de los datos y patrones para escribirlas en lenguaje natural estructurado y lenguaje técnico (LPT), y se formulan los pasos para transformar las reglas de LPT a recursos de bases de datos. Además se propone una arquitectura de software para administrar dichas reglas en bases de datos relacionales y una aplicación concreta: LPT-SQL. Finalmente se incluyen dos estudios de casos y los resultados de la evaluación de expertos sobre el conjunto de categorías y su expresión en LPT para permitir su implementación automática.

ABSTRACT

The business rules approach is a subject-matter in which the rules are administered by certain independence of the information systems and it allows the automation of the same ones.

There are rules which are represented in the same scheme of the database, it doesn't happen in this way with those associated to the behavior of the data and their operations.

The automatization generally occurs to the identification of the rules based on rules categories from two points of view: business and IS, expressed on the abstraction levels: informal, technical and formal (implementation). The use of engines of rules for this automatization is popular, but argued. As a result of this paper a set of categories from the perspective of the data and patterns to write them in a natural language is proposed; a language of technical patterns, LTP, is created to express them using the business terms, persistent in relational databases. The steps to transform the LTP rules to databases resources are formulated. Moreover, a software architecture to implement in an automatic way the business rules in relational databases and one concrete application, LTP-SQL, are proposed.

Finally, two case studies and the outcomes of experts' assessment about the set of categories of proposed rules and their LTP expression to achieve the automatic implementation are included.

Índice

Pág.

INTRODUCCIÓN

La descripción de los requisitos juega un papel esencial en el ciclo de vida de un sistema de información (SI), en que las reglas de negocio son consideradas ciudadanas de primera clase según el Grupo de Reglas de Negocio (Business Rules Group) [1], pues reflejan la forma en que las empresas hacen los negocios, y actúan como un componente clave en el diseño de un SI. Si las reglas cambian el SI debe transformarse de alguna manera.

Las reglas de negocio han sido tradicionalmente implementadas por el programador como parte del código fuente de las aplicaciones, en los objetos de alguna base de datos, o en ambas; con un consiguiente costo de codificación y mantenimiento.

Desde mediados de la década de 1980 se trabaja en expresar explícitamente las reglas de negocio [2] y contribuir a su implementación en los SI de manera que garanticen su efectivo cumplimiento [3].

Actualmente es una necesidad creciente de los negocios que sus políticas sean reflejadas en sus SI de manera menos costosa y más eficiente, con una menor implicación de los programadores en la implementación de las mismas, de modo que cambios en las reglas no implique contratar servicios de mantenimiento de los sistemas [4].

Un componente clave de la generalidad de los SI son las bases de datos (BD) cuyos esquemas conceptuales y lógicos permiten captar algunas reglas de negocio inherente a los mismos [5, 6]. Otras reglas, aunque no se hayan representado en dichos esquemas [7-9], tienen mucha relación con los datos del negocio y pueden quedar implementadas dentro de la BD, generalmente asociadas con las operaciones CRUD (siglas en inglés de las operaciones crear-Create, leer-Read, actualizar-Update, eliminar-Delete), sobre los datos [10, 11].

Prácticamente todos los autores que tratan las reglas de negocio emplean diversas categorías o clasificaciones para las mismas. Se aprecia que las clasificaciones más comunes se obtienen considerando la perspectiva de los SI, la perspectiva de negocios, o ambas [12-19]; sin embargo, no se ha encontrado en la literatura revisada una clasificación de reglas de negocio desde la perspectiva del comportamiento de los datos del negocio, útil para el desarrollo de SI que manejen alguna BD.

Por otro lado, para representar las reglas de negocios en base a sus categorías se reconoce el uso de patrones [20-23]. Para dicha representación se han propuesto varios formalismos y niveles de expresión [17, 22].

El nivel llamado informal o versión en lenguaje natural permite que las reglas sean entendibles por todos los implicados: los especialistas del negocio y los especialistas de sistemas ya sea arquitecto, analista, etc. Se observa con frecuencia la definición de un patrón de regla para cada categoría usando lenguaje natural. El nivel técnico o versión en lenguaje de especificación de reglas, dirigido a los especialistas del sistema, representa un nivel intermedio entre lo informal y lo formal; en este sentido algunos autores utilizan lenguajes basados en XML [21, 24-26], otros lenguajes como OCL (lenguaje de restricción de objetos, de sus siglas en inglés Object Constraint Language) que permiten expresar restricciones en el contexto del diseño de clases soportadas sobre UML (lenguaje unificado de modelado, de sus siglas en inglés Unified Modeling Language) [27]. Sin embargo, el uso de cualquiera de estos lenguajes para escribir las reglas en el nivel técnico, puede hacer complejas las expresiones en el contexto de tablas, atributos e interrelaciones del esquema relacional de la base de datos del negocio; crear un lenguaje con la simplicidad necesaria y la riqueza semántica para escribir las reglas a partir de la base de datos, puede ser una buena solución. Los lenguajes formales, aceptados como aquellos que permiten implementar las reglas ejecutables sobre computadoras, están basados en lenguajes de programación o cualquier tipo de recurso computacional.

El *enfoque de reglas de negocio* [28] plantea que las reglas identificadas y escritas con los formalismos creados al efecto, se gestionen con cierta independencia de los demás elementos del SI, y su código se genere automáticamente; deja atrás la manera tradicional de codificarlas por los programadores del sistema.

Existen así diversos tipos de herramientas de software dirigidas a la implementación automática de las reglas. Los Sistemas de Gestión de Reglas de Negocio (SGRN) o BRMS (de sus siglas en inglés Business Rules Manager System), que contienen como núcleo un motor de reglas de inferencia, ayudan a implementarlas con el *enfoque de reglas de negocio*, permitiendo que los propios expertos del negocio participen en la escritura de las reglas para los SI, lo cual parece una buena solución; sin embargo no lo es realmente en la práctica [29]. Las desventajas principales de utilizar los SGRN, reconocidas por [30] y apoyadas por [31], son las siguientes:

- Se requiere de alta experticia para desarrollar e implementar con un SGRN específico, además de habilidades avanzadas en software para diseñar y desarrollar aplicaciones, así como de comunicación efectiva usando las mejores prácticas; por todo lo anterior requiere un incremento en los costos de recursos humanos.

- Normalmente los ciclos de desarrollo son extensos.
- Un SGRN no es un ambiente de desarrollo totalmente adecuado, pues su eficiencia decrece ante tareas complejas.
- Los cambios de su modelo de objeto pueden tener implicaciones importantes para el sistema; si el modelo cambia, esto afectará a todas las reglas cuyas condiciones y acciones dependen de estas. Los cambios de requerimientos de software puede llevar horas de trabajo para tratar errores de compilación. Esto sucede, sobre todo, porque los datos son enviados al motor de reglas a través de un modelo de objeto.
- El desarrollo con un SGRN conlleva a un fuerte acoplamiento con un vendedor específico, y una vez que se ha tomado la decisión de desarrollar un sistema usando un SGRN determinado, el éxito de la implementación está en la funcionalidad y atributos del SGRN particular. La opción de eliminar el uso del producto conduce a un serio costo de desarrollo, con implicaciones significativas de tiempo.

Ya en [32] se había abordado también sobre el error de usar exclusivamente los motores de reglas, pues algunas de ellas pueden ser implementadas de manera más simple fuera de estos. También se develan los comportamientos ineficientes de los SGRN cuando es necesario un tratamiento intensivo de los datos [33].

Puede afirmarse que son escasas las herramientas que permiten la generación automática de las implementaciones de reglas sobre las bases de datos relacionales [34-36].

Herramientas como el Dresde OCL Toolkit, poseen motores de transformación de reglas de negocio en código de consulta o DQL (del inglés Data Query Language); la transformación se centra en generar de manera automática las reglas concebidas de forma conceptual, se construye una plataforma de generación de código para consultas de integridad [37]. Otras herramientas ejecutan reglas como recursos SQL, aceptan la entrada de reglas de solo una categoría, expresadas en algún lenguaje específico. Por ejemplo, Athenas descrito en [21, 38] acepta solo reglas de tipo restricción, escritas en OCL que se integran con Oracle.

Las aplicaciones estudiadas que generan las reglas como recursos de BD se centran en generar solamente restricciones de integridad.

En el contexto en que se enmarca esta investigación, el proceso de informatización de la sociedad cubana, es muy usado el modelo relacional para trabajar con los datos del negocio que deben

persistir en una base de datos, accesible por diversas aplicaciones encargadas de ayudar al control de cualquier negocio.

Los planteamientos anteriores conducen al siguiente *problema científico*:

¿Cómo clasificar, especificar por niveles de expresión e implementar las reglas de negocio teniendo en cuenta el comportamiento de los datos para base de datos relacionales, prescindiendo de los motores de reglas de inferencia y de la programación tradicional?

Para abordar este problema se formulan las siguientes preguntas de investigación:

1. ¿Cómo clasificar las reglas de negocio que se distinguen por su relación con el comportamiento de los datos del negocio y se pueden reconocer como reglas desde una perspectiva de los datos?
2. ¿Cómo expresar las reglas de negocio desde la perspectiva de los datos a partir de las relaciones y atributos especificados en el esquema lógico de la base de datos?
3. ¿Qué estrategia se define para determinar las relaciones o tablas en las cuales deben ser chequeadas o evaluadas las reglas?
4. ¿Cómo generar automáticamente las implementaciones de las reglas de negocios, de las categorías con perspectiva de los datos, como mecanismos de las bases de datos relacionales?

Teniendo en cuenta lo anteriormente planteado, el presente trabajo está dirigido hacia el cumplimiento del siguiente *objetivo general*:

Proponer un conjunto de categorías de reglas de negocio desde la perspectiva de los datos, los formulismos para su expresión y los elementos necesarios para generar automáticamente su implementación en forma de recursos SQL, dentro del esquema físico de una base de datos relacional.

Para cumplir este objetivo general se formulan los *objetivos específicos* siguientes:

1. Definir un conjunto de categorías de reglas de negocio desde la perspectiva de los datos, haciendo corresponder un patrón a cada categoría.
2. Definir la sintaxis y la semántica de un lenguaje de nivel técnico para escribir los elementos de los patrones de reglas, a partir de las relaciones y atributos que se corresponden con los objetos del negocio.
3. Establecer un conjunto de pasos para realizar la traducción de las reglas de negocio del lenguaje técnico (LPT) a lenguaje SQL, como mecanismos de bases de datos relacionales.

4. Desarrollar una herramienta de software que permita traducir las reglas escritas en lenguaje técnico a recursos de bases de datos relacionales.
5. Evaluar el conjunto de categorías de reglas y las formas de expresarlas a través de los procedimientos definidos en la investigación.

Justificación de la investigación

El enfoque de las reglas de negocio y su implementación se establecen en función de cómo se representan, implementan y ejecutan las reglas. Para lograr la implementación de las reglas dentro de las bases de datos, se deben tratar aspectos como las categorías de reglas que se utilizan, cómo se representan, cuáles recursos se usan para su implementación y cuándo se ejecutan.

La literatura revisada muestra diversidad de conjuntos de categorías de reglas de negocio para clasificarlas; sin embargo, no se abordan directamente tipos de reglas en función de los datos del negocio, por lo que se percibe la utilidad de considerar un conjunto de categorías de reglas de negocio, en forma de patrones, que permitan expresar las reglas desde una perspectiva de los datos.

Con esta perspectiva las reglas pueden expresarse en un lenguaje que permita su formalización en función de las relaciones y atributos de la base de datos, para traducir estas a recursos o mecanismos para su implementación en BD relacionales, lo cual facilita la representación de los requerimientos del negocio en los SI.

A partir del planteamiento del problema y como resultado del análisis de la literatura especializada se formuló la siguiente **hipótesis**:

> La conformación de un conjunto de categorías de reglas de negocio desde la perspectiva de los datos y las formas de expresarlas permiten su implementación automática dentro de la base de datos del negocio en un contexto relacional.

Por todo lo anterior el trabajo que se propone tiene valor teórico, práctico y metodológico.

Su **valor teórico** está dado por:

La definición de un conjunto de categorías de reglas de negocio desde la perspectiva de los datos, los patrones para expresarlas en lenguaje natural estructurado y lenguaje técnico, así como una estrategia para generarlas dentro de bases de datos relacionales, constituyen los fundamentos

teóricos de una plataforma de trabajo que permite la implementación y mantenimiento de las reglas de manera automatizada.

El **valor práctico** consiste en la obtención de una herramienta de software para la implementación automática de reglas de negocio en bases de datos, que ha sido utilizada en SI operacionales, entre ellos: la base de datos para el sistema de control de trasplante renal del hospital provincial de Santa Clara "Arnaldo Milián Castro"; la planificación del menú en los comedores de la Universidad Central "Marta Abreu" de Las Villas (UCLV); y el control de evaluaciones en la secundaria básica urbana de Santa Clara "Héctor Martínez Valladares".

Su **valor metodológico** consiste en que el conjunto de categorías de reglas desde la perspectiva de los datos y los patrones propuestos guían la identificación de las reglas de esta naturaleza, así como el resto del proceso, para lograr su implementación automática.

La novedad científica de la investigación está dada por:

1. Sistematización de un conjunto de categorías de reglas de negocio desde la perspectiva de los datos y la proposición de los patrones correspondientes en el lenguaje técnico LPT, que conforman un repositorio de reglas de negocio cuyos elementos utilizan relaciones y atributos de la base de datos de un negocio.
2. Obtención de un conjunto de pasos, para traducir las reglas de negocio desde el lenguaje técnico LPT a SQL como lenguaje formal, para finalmente expresar las reglas mediante recursos de bases de datos relacionales.

El documento de la tesis ha sido estructurado en tres capítulos:

En el **primer capítulo** se presentan los referentes teóricos que se consideran relevantes acerca de las reglas de negocio; diferentes clasificaciones de reglas, dadas por varios autores; los distintos niveles de expresión de las mismas, así como lenguajes y formalismos existentes para escribirlas. Además, se presentan lenguajes para lograr la interoperabilidad de las reglas, así como las diferentes formas que básicamente se han utilizado para implementarlas.

El **segundo capítulo** se inicia con la presentación del conjunto de categorías de reglas propuestas en este trabajo, sus patrones correspondientes y luego se describe el lenguaje LPT (lenguaje de patrones técnico), creado para escribir las reglas en un nivel técnico basado en las relaciones (tablas) y atributos. Se presenta una manera de traducir las reglas de LPT a lenguaje SQL, como mecanismos de bases de datos, de manera que se logran determinar las tablas donde se evalúan o se hacen cumplir las reglas.

El **tercer capítulo** se dedica a la descripción de los recursos SQL propuestos para la implementación de cada tipo de regla; se propone una manera de implementar el control de versiones para la modificación de las reglas, específicamente las de restricción. Se analiza la repercusión en los datos de las modificaciones de las reglas de restricción y se establece un conjunto de pasos a seguir para que una herramienta de software trate las modificaciones de esta categoría de reglas. Se presentan procedimientos de validación acerca de la utilidad del conjunto de categorías de reglas desde la perspectiva de datos y de su escritura en lenguaje técnico de patrones (LPT), para su posterior implementación automática. También, se describe la herramienta de software obtenida para generar las implementaciones de las reglas de negocios desde la perspectiva de los datos y una forma de validarla.

Finalmente se incluyen las **conclusiones** que resaltan los principales resultados obtenidos en la investigación y las **recomendaciones** de aquellos aspectos que se consideran deben dársele continuidad.

1. REGLAS DE NEGOCIO: CATEGORÍAS, LENGUAJES E IMPLEMENTACIONES

Este capítulo tiene como objetivo presentar un conjunto de consideraciones que aparecen en la literatura especializada sobre las reglas de negocio, sus categorías, así como formalismos para expresarlas y maneras de implementarlas. Se muestra la tendencia de identificar las reglas explícitamente y tratarlas con cierto grado de independencia de los SI, de modo que sean creadas y modificadas teniendo en cuenta los principios de versiones e historial de reglas, asimismo se realiza una valoración crítica de estos aspectos.

1.1. Las reglas de negocio

Una regla de negocio según Ronald G. Ross, conocido como el padre de las reglas de negocio, es una regla que está bajo jurisdicción del negocio, lo cual significa que puede ser creada, revisada y eliminada cuando el negocio lo estime conveniente [39, 40].

Los requisitos reflejan las políticas o reglas de negocio y han sido tradicionalmente implementados por el programador como parte del código fuente de las aplicaciones, o en los objetos de alguna base de datos, o en ambos; con un consiguiente costo de codificación y mantenimiento.

Los precedentes históricos de tratar las reglas de negocio con especial distinción se refieren a la década del 80 [2, 3]. Con el objetivo de gestionar las reglas de forma sistemática y centralizada se desarrollaron aplicaciones de intérpretes simples usando el algoritmo RETE y así se iniciaron los productos comerciales de máquinas de reglas de negocio: System Corporations G" y Hley Systems en 1986, Blaze Advisor de 1988, así como ILOG SA de 1989 [3].

Según se define en [41], "una regla de negocio es una sentencia que define o restringe algunos aspectos del negocio. Tiene la finalidad de establecer la estructura del negocio, controlar o influenciar el comportamiento de negocio […] generalmente son estudiadas desde dos perspectivas: desde la perspectiva del negocio, que pertenece a cualquier restricción de la conducta de las personas en la empresa y desde la perspectiva de los SI, relacionadas con los hechos que son grabados como datos y las restricciones sobre los cambios a los valores de tales hechos, además de los procesos que hay que seguir".

Las reglas de negocio escritas como sentencias permiten a los usuarios expertos definir políticas, condiciones, modelar el conocimiento del negocio en unidades pequeñas y aisladas de otros requerimientos [22].

El término "enfoque de reglas de negocio" (BRA, acrónimo del inglés: Business Rules Approach) se emplea con frecuencia para referirse al manejo de estas reglas, independientemente de las aplicaciones que las hacen cumplir; de acuerdo con este enfoque, las reglas de negocio son relativamente independientes del resto de los requerimientos de un SI y necesitan un tratamiento especial [38].

Entre las propiedades deseables en las declaraciones de reglas están la atomicidad, no ambiguas, compactas, consistentes y compatibles [22].

Varios autores coinciden en que las reglas de negocio tienen un significativo efecto sobre la adaptabilidad y flexibilidad de las aplicaciones [22, 42-45]. Otros beneficios derivados del uso de reglas de negocio muy importantes, según [14] son: agilidad, reducción de costo y transparencia en la creación de los SI.

En [22] se enfatiza en un conjunto de características fundamentales que las reglas de negocio deben cumplir y que la autora de este trabajo también reconoce; estas se resumen a continuación:

- En términos generales, son restricciones que definen condiciones, las que deben ser verdaderas en situaciones especificadas, bajo las cuales un proceso es realizado, o que existirán después de completado un proceso.
- Las reglas de negocio definen lo que debe ser, más bien el "qué hacer" que el "cómo hacer".
- Las declaraciones de reglas de negocio en el modelo de negocio definen la lógica deseada del negocio, ellas describen una situación que el negocio exige; si se expresan como funciones lógicas, siempre devolverían el valor "verdadero".
- Desde una perspectiva lógica de reglas de negocio, no hay excepciones, algunas reglas pueden parecer obvias, pero si es así, esto es bueno porque se ha construido una lógica de negocio comprensible.
- El analista de negocio debe especificar una serie de declaraciones claras sobre la lógica base de un negocio.

El aspecto de la claridad, es crucial; deben ser tan claras que el experto o dueño del negocio las acepte inmediatamente como válidas o las rechace como inválidas. El analista debe comprender

que las personas del negocio no entienden de lógica de un lenguaje matemático y que no tienen una perspectiva de tecnología, pero ellos pueden hacer que el negocio funcione usando solamente declaraciones simples, que el analista debe captar. Así, el verdadero poder de las reglas está en:

- La capacidad de hacer declaraciones de nivel de negocio que pueden ser traducidas de un modo casi directo a un sistema operacional.
- El efecto combinado de un número relativamente grande de declaraciones simples, de modo que juntas, tienen un impacto que es mayor que la suma de las partes individuales.

1.2. Categorías para reglas de negocio

Debido a la diversidad y complejidad de las reglas de negocio los autores tienden a agruparlas y clasificarlas siguiendo diferentes perspectivas, siempre con el objetivo común de organizar todas las reglas del análisis de requisitos del negocio.

En ocasiones los propietarios de negocios y los desarrolladores no coinciden en la manera de tratar las reglas [11]. Desde el punto de vista de los SI se debe establecer cuál es origen de las reglas: (1) traducciones de reglas de negocio en sus implementaciones y (2) creación de nuevas reglas provenientes del diseño del sistema. En [44] se denomina a estas últimas reglas de sistema, para distinguirlas de las reglas de negocio propiamente.

Las clasificaciones (conjunto de categorías) que combinan las cuestiones empresariales con las características técnicas del sistema tienden a obstaculizar la trazabilidad entre los SI y las necesidades del negocio, que puede dar lugar a paradojas que son difíciles de conciliar [46].

Un aspecto crucial del negocio es el manejo de sus datos, y este es la base de la motivación fundamental de esta investigación: la administración de las reglas de negocio que tienen su acción directa en bases de datos relacionales. Se reconoce así la necesidad de indagar sobre clasificaciones de reglas de negocio centradas en el comportamiento de los datos.

La revisión de literatura especializada evidencia que varios autores al definir conjuntos de categorías de reglas, sin precisarlo; definen subconjuntos de estas vinculadas a los datos que se manejan en el negocio, esto es natural porque según se expresa en [10] los datos y los sistemas de bases de datos son repositorios centrales de información para los negocios.

A continuación se describen diferentes categorías de reglas de negocio comunes a varios autores, ellos se refieren a reglas que surgidas en el negocio afectan el comportamiento de los datos del negocio en sí mismo y por tanto repercutirán en su representación digital, en bases de datos. Para

cada categoría analizada se exponen los autores que las defienden y las características generales de su funcionalidad.

Las reglas de restricción son incluidas por Date [47] Solivares [13] Morgan [22], Ashwell [12], Lowenthal [14], Ross [48], Von Halle [17], Ioana Matei [15] y otros. En todas ellas son condiciones que deben ser cumplidas por los datos para que se consideren válidos. Andreescu [49] presenta además las reglas de enumeración y las reglas de integridad para capturar restricciones que se aplican a los atributos de una entidad.

Matei [15] reconoce las líneas directivas o restricciones suaves, que pueden ser interpretadas como sugerencias.

Según la concepción de SBVR (por las siglas en inglés de Semantic Business Vocabulary and Rules), las reglas de acuerdo a su formulación lógica se clasifican en reglas estructurales y reglas operacionales o de obligación. De las categorías anteriores, las estructurales especifican las restricciones de integridad que no pueden ser violadas.

En [22] se declara que en cierto modo las reglas de restricción se solapan con las reglas del modelo de datos, pues también impiden la introducción de datos erróneos, pero la diferencia estriba en que las reglas de restricción condicionan el valor de los atributos o propiedades de una entidad con más especificación que las restricciones básicas existentes sobre las mismas en el modelo de datos. En el propio trabajo se reconocen las reglas de restricción básica y la lista de restricciones.

Las reglas de cómputo o cálculo son reconocidas por Ross,Von Halle [17, 50], Morgan [22] y Matei [15]. Una regla de cómputo es una declaración completa que provee operaciones para lograr el valor de un término, donde los operadores pueden incluir suma, diferencia, producto, cociente, conteo, mínimo, máximo y promedio. El resultado del cómputo puede ser un valor nuevo para un atributo.

Ross se refiere a reglas de definición, para organizar el conocimiento básico del negocio contribuyendo al significado de conceptos, y centrándose en la esencia del propio negocio [16, 40], llama reglas de derivación a las reglas que permiten hacer la definición de términos a partir de los existentes. Andreescu [49] presenta la categoría *if…then*, y las usa para definir o clasificar un término del negocio como subclase de otro.

Por otra parte se aprecia en las categorías de reglas de negocio, asumida por Zoet [19], que usa algunas asociadas directamente a los datos, a estas reglas les llama reglas de condición de datos, y reglas de control de resultados.

Date [5] se manifiesta a favor de que las reglas sean parte de las bases de datos; para básicamente garantizar la integridad de los datos; sin embargo defiende las ventajas de mantener las reglas separadas del motor de bases de datos [5, 47], almacenadas en un repositorio central, para permitir su mantenimiento independientemente de los datos. Las reglas cerca de los datos[1] pueden tener incidencia más directa sobre estos y en las funcionalidades CRUD [11] que son comunes a todos los gestores de datos. Es esencial considerar que uno de los aspectos básicos en el trabajo con reglas de negocio y los SI es lograr la satisfacción de las necesidades de la organización con el sistema [46].

Como resultado del estudio realizado sobre diversas categorías de reglas presentadas por diferentes autores se observa que existen tipos de reglas que están directamente relacionadas con los tipos de entidades del negocio y sus correspondientes propiedades, por tanto con los datos del negocio; sin embargo ninguno de los autores estudiados conforma una clasificación en la que cada clase o categoría esté definida en función de los datos del negocio.

1.3. Maneras de expresar las reglas de negocio y lenguajes para escribirlas

Los niveles para expresar las reglas son cuatro según [17], cada uno para una audiencia diferente: conversación informal del negocio, versión en lenguaje natural, versión en lenguaje de especificación y versión en lenguaje de implementación. En última instancia las reglas se traducen del lenguaje de especificación al lenguaje de implementación, el cual tiene todo el potencial para ser ejecutado [37, 38, 51].

En [22] se distinguen tres niveles de expresión de reglas de negocio: *informal*, con la regla de negocio expresada como sentencia en lenguaje natural, tal y como el cliente del negocio desee; *técnico*, en que se combina referencias a datos estructurados, operadores y restricciones con el lenguaje natural; y el *formal*, de implementación, para proporcionar sentencias conforme a una sintaxis definida y proporciona la funcionalidad completa de la regla.

En este trabajo se utilizan los niveles expuestos por Morgan [22], reconociendo que el nivel técnico se corresponde con el de especificación, así como el formal al de implementación.

[1] Cerca de los datos se refiere a implementada junto a los datos.

Existen diversos lenguajes para representar reglas de negocio, pero comúnmente no se presentan de antemano clasificados en uno de estos niveles.

En algunos de los formulismos las reglas de negocios son expresadas en forma de patrones. Estos patrones usan frases claves para expresar las reglas de manera estructurada. Entre ellos se publican RuleSpeak [20], los patrones de Weiden [52], los patrones de Morgan [22].

RuleSpeak® es un conjunto de pautas para expresar reglas de negocio de forma concisa que facilita la redacción de la regla para cada una de las partes involucradas en el negocio; es desarrollado por Ross, publicado por el Object Management Group (OMG) en 2007.

Weiden propone diecisiete patrones de reglas, agrupados en tres categorías [23]: estructura de conceptos, persistencia e historia, para las reglas de categoría estructural. Las primeras se asocian a los conceptos del negocio (que son instanciados en los datos del negocio) y las relaciones existentes entre ellos. Las de persistencia e historia se vinculan a los datos del negocio que deben ser almacenados a través del tiempo.

Los patrones, según Morgan, consideran de una manera u otra a algún sujeto, sus características y hechos, así como la existencia de un modelo de hechos que reconozca términos y frases aceptadas para el dominio del negocio.

En [22] se propone el uso de la notación punto —del inglés Dot Notation– para eliminar la ambigüedad del lenguaje natural para expresar las reglas del negocio. La notación punto, familiar a los programadores, y su alternativa Of-Notation son propuestas de Halpin [53, 54] para expresar las relaciones entre los conceptos, para ayudar en la navegación a través de las clases del modelo del negocio. Esto deriva en una mejor claridad [55] en la expresión de reglas complejas.

OCL, basado en notación punto, se ofrece como lenguaje para la especificación de restricciones, incorporado a la versión 1.1 del UML [56]. Con OCL se complementan los aspectos que no se pueden expresar con los diagramas UML, como son las restricciones; también permite hacer aclaraciones de ambigüedades.

OCL puede ser usado con distintos propósitos: como lenguaje de navegación y de consulta, para especificar restricciones sobre operaciones así como describir pre y post condiciones de operaciones y métodos, según Bruegge [57] y [27, 58]. Sus funcionalidades son reflejadas en algunas herramientas CASE [59, 60]. En [22] se reconoce su utilidad pero no lo utiliza, declara que su uso carece de entusiasmo por su sintaxis desalentadora para los expertos del negocio y también la mayor parte de los analistas.

Los Sistemas de Gestión de Reglas de Negocio (SGRN) usan una variedad de lenguajes propietarios para representar las reglas de negocio que gestionan.

Actualmente se aprecian esfuerzos de estandarización para representar estas reglas de la forma *if…then* que garanticen el intercambio de reglas en la Web. Entre estos lenguajes se destacan RuleML (Rules Markup Language) [24], el SWRL (Semantic Web Rules Language), RIF (Rules Interchange Format*)* [21], [25], PRR (Production Rules Representation) [26], que han surgido en la búsqueda de un lenguaje común para motores de reglas con algoritmos de encadenamiento hacia adelante, para tratar las reglas expresadas como *if…then*. Estos lenguajes tuvieron sus principios en la iniciativa CommonRules de IBM proseguidos por el BRML (Business Rules Markup Language) [61] y SRML (Simple Rules Marckup Language) [62].

El SBVR (Semantic Business Vocabulary and Rules) [63, 64] proporciona una forma estándar para crear modelos de vocabulario, conceptos y formulaciones semánticas; pero no es un lenguaje que las personas del negocio utilicen para escribir reglas, es más bien un metalenguaje. Bajo la guía de SBVR el personal del negocio no requiere conocimientos o habilidades informáticas para definir las reglas de su negocio, pero no ofrece ni un modelo listo para ser transformado en software, ni un lenguaje de modelación a ser implementado [65, 66].

A continuación se relacionan aspectos esenciales de estos lenguajes basados en XML en la tabla 1.1 comparativa a partir de resultados de [21] y elaboración propia.

Tabla 1.1 Comparación de lenguajes para reglas de negocios basados en XML.

Lenguaje	**Organismo de estandari-zación**	**Nivel de abstracción**	**Función de la especificación**	**Tipos de reglas que representan**	**Herramien-tas disponibles**
RuleML		Medio	Familia de lenguajes. Esquemas XML para intercambiar reglas en la Web con alta precisión	Reglas de deliberación: derivación y modal *y* las reglas de reacción: procesos de eventos complejos Forma if… then	Implementa ciones (jDrew, Mandarax)
SBVR	OMG	Alto (metalenguaje)	Captura semántica de un conjunto de significados compartidos	------	Easyfiller [67]
SWRL	W3C	Medio	Combina OWL, DL y OWL Lite. Ayuda al intercambio de datos y significados entre servicios Web. Orientado a integración con ontologías .Recoge los conceptos a los que se hace referencia en las reglas de negocio	Reglas de la forma de implicación entre un antecedente y un consecuente. Puede expresarse en formato XML o RDF	Editores y razonadores (Protegé)
RIF	W3C	Medio	Pretende asegurar interoperabilidad entre tecnologías basadas en reglas. Permite consistencia con respecto al resto de las especificaciones. Orientado a integración con ontologías. Recoge los conceptos a los que se hace referencia en las reglas de negocio	Reglas de la forma de implicación entre un antecedente y un consecuente.	Varias implementa ciones Protegé
OMG-PRR	OMG	Bajo	Representa reglas de producción	Reglas de la forma if… then	---

Fuente: [21] y Elaboración propia

Algunos de estos lenguajes se presentan muy ligados al nivel de implementación y el nivel de abstracción baja de OMG-PRR [26] se debe a que asume determinados aspectos del entorno tecnológico. Un nivel de abstracción alto implica que no están relacionados con la implementación. El metalenguaje SBVR es completamente independiente de su implementación. Actualmente son escasas las herramientas para trabajar con él (aparte de las guías proporcionadas en el documento que define el estándar).

Los lenguajes SWRL y RIF, están especialmente orientados a la integración con ontologías. Éstas recogen los conceptos a los que es posible hacer referencia en las reglas de negocio. Herramientas como Protegé disponen de extensiones para la especificación de reglas de negocio expresadas en SWRL.

Un resumen general entre las diferentes maneras de expresar las reglas de negocio, entre los diferentes tipos de patrones estudiados, el lenguaje OCL y los lenguajes basados en XML se muestra en la tabla 1.2.

En dicha tabla se revela cada grupo de lenguajes con sus funciones establecidas, los lenguajes basados en XML que generalmente representan un antecedente y un consecuente de una regla que se puede leer como *if…then*, se hacen realmente útiles para intercambiar reglas de este tipo en ambiente Web. Se observa que ninguno de estos tipos de lenguajes puede ajustarse fácilmente con claridad y sencillez para escribir reglas en función de los términos del negocio representados en las tablas y atributos de las bases de datos de los negocios.

Tabla 1.2 Comparación de formalismos para expresar reglas de negocio.

Formas de expresión	**Basado en**	**Categoría**	**Nivel de abstracción**	**Herramientas**	**Aportes**	**Limitaciones**
Rulespeak	Patrones	Derivación, cómputo, inferencia, restricción	Alto-lenguaje semiestructurado	No se conocen	Cercano al personal del negocio	Dependientes de las categorías
Weidem	Patrones	Estructura, comportamiento administrativo	Alto-lenguaje semiestructurado	No se conocen	Cercano al personal del negocio	Dependientes de las categorías
Morgan	Patrones	Restricción básica, listas de restricciones, clasificación, enumeración	Alto-lenguaje semiestructurado	No se conocen	Patrones de negocio comprometidos con la automatiza-ción	Dependientes de las categorías
Otros	Patrones	Diversas categorías, plantillas	Alto	Jboss Rules. Drool, Ilog Jrules	Cercano al personal del negocio	If…then
OCL	Seudocódigo, restricciones sobre objetos,	Restricciones	Técnico-Medio	OCL Tool Kit Athenas	Especifica restricciones en un modelo orientado a objetos, complementa UML	Se basa en el modelo OO, solo para restricciones
XML	Código XML	If…then	Medio, Alto, Bajo	RuleML, Protege, Drew	Intercambio en la Web	Complejo para expresar relaciones entre objetos y tablas de base de datos

Fuente: Elaboración propia.

1.4. Implementación de las reglas de negocio

En [68] se afirma que, las reglas de negocio pueden estar en diferentes capas de la aplicación: en el cliente, en la capa de datos, en la capa media.

Existen disímiles formas de implementar las reglas, incluso pueden existir varias técnicas para implementar una misma regla [68]. Al considerar cada una de las alternativas para determinar cuál funciona mejor en una determinada situación, se deben tener en cuenta: la viabilidad a largo plazo de la estrategia, el rendimiento en tiempo de ejecución, el grado de cumplimiento de la regla, la flexibilidad, y la capacidad para mantener las operaciones del negocio.

Entre las formas de implementación [32] se destacan tradicionalmente: sentencias de programas a través de código en línea [42, 69], secuencias de comandos, componentes especializados de reglas, mecanismos de bases de datos [10, 70, 71]; se reconocen los chequeos (del inglés check) [72], los disparadores [73-75], los procedimientos almacenados [75], vistas [73]. Existen métodos más modernos [32] como: los motores de reglas [3, 44, 76], los sistemas de flujo de trabajo con un motor interno y estilos personalizados para las reglas de codificación [77], y las tablas de búsqueda. Cada una de estas alternativas presenta sus características, ventajas y desventajas.

Se evidencia una fuerte tendencia a implementar las reglas automáticamente usando los motores de reglas o los SGRN como una buena solución basada en BRA [30, 78-86], sin embargo Morgan apoya la posibilidad de que algunos tipos de reglas [32], se pueden implementar de manera más simple o mejor fuera de estos. Actualmente se destaca un grupo de desventajas de los motores de reglas y específicamente de los SGRN que deben tenerse en cuenta[30, 31] entre las que se destacan:

- Utilizarlos requiere alta experticia de los desarrolladores e implementadores, con un SGRN específico, así como habilidades avanzadas en el diseño de software, desarrollo aplicaciones y altas habilidades de comunicación efectiva usando las mejores prácticas, por tanto requiere personal costoso.
- Los ciclos de desarrollo generalmente son extensos por lo que se hace un significativo uso de tiempo.
- No son óptimos para muchas tareas computacionales, pues con un SGRN la eficiencia decrece ante tareas complejas, se ha fracasado en los intentos de hacer plantillas de código, pues esto es difícil para los departamentos de tecnología de la información y más aún para los desarrolladores de reglas.

- Los cambios del modelo de objeto puede tener implicaciones importantes para el sistema, si el modelo cambia esto afectará a todas las reglas cuyas condiciones y acciones dependen de estas. Cambiar un requerimiento de software puede llevar horas de trabajo para fijar errores de compilación. Esto sucede sobre todo pues los datos son enviados al motor de reglas a través de un modelo de objeto.
- El desarrollo con SGRN conlleva a un fuerte acoplamiento con un vendedor de SGRN
- El éxito de la implementación está en la funcionalidad y atributos del SGRN particular con el que se ha tomado la decisión de desarrollar un sistema, la opción de eliminar el uso del producto lleva a un serio costo de eficiencia y desarrollo, con implicaciones significativas de tiempo.

Tabla 1.3 Comparación entre formas de implementación de las reglas de negocio.

Formas de implementar	**Viabilidad**	**Rendimiento**	**Mecanismos Evaluación**	**Trazabilidad**	**Tiempo y costo**	**Flexibilidad y capacidad**	**Categorías de reglas de negocio que permite implementar**
Sentencias de Programas	Media. Se mejora encapsulando reglas en funciones	Alto	Seleccionar entre ramas de código alternativo, basado en una condición dada	Baja	Tiempo Significativo para programar la regla Alto costo ante los cambios.	Alto	Todas
Script	Viable. Media	Bajo	Ubica el código de la regla fuera del código del programa. Si se quieren cambiar los límites a una regla, sólo se necesita cambiar el script.	Media	Tiempo significativo para programar la regla Alto costo ante los cambios	Media	Todas
Componentes	Viable. Media.	Alto	El código dentro del componente de regla está escrito para satisfacer un conjunto limitado de objetivos, un problema específico. Funcionalidad disponible a través de interfaces publicadas	Media.	Costo de invocación alto	Media	Todas

Mecanismos de BD	Viable Media	Medio	Disparadores y recursos de BD	Media	Medio	Bajo	Restricción ante funcionalidades CRUD
Motores de reglas de inferencia	Viable Alta	Medio Bajo cuando BD grandes	Algoritmo Rete	Alta	Inversión significativa de tiempo para aprender a usar: Bajo costo ante los cambios	Alta	If … then
Motores de flujos de trabajo	Viable. Alto	Alto	Creación y seguimiento de instancias de procesos, definidos en una estructura que puede controlar la actividad relevante casi en tiempo real	Alta	Puede hacer cambios casi en tiempo real	Alta	Reglas sencillas que expresen flujo de trabajo
Tablas de búsqueda	Viabilidad Baja. Uso de tablas sin reflejar las sentencias de reglas. Difíciles de mantener en ocasiones	Alto.	Una o más reglas de negocio están codificadas en una tabla. Se necesita definir qué tipo de evento, en qué proceso afecta al negocio y si produce un impacto Se evalúan implicaciones lógicas	Media	Bajo Implementación rápida	Baja Pueden ser difíciles de mantener. Las entradas no son absolutamente evidentes.	Expresar la lógica de negocio en una forma *if…then*

Fuente: [32] y Elaboración propia.

Al comparar las diferentes formas de implementación de reglas de negocio a primera vista puede parecer que la mejor forma propuesta es a través del uso los motores de reglas de inferencia, pues los parámetros viabilidad, trazabilidad y la flexibilidad son altos. En el caso de la flexibilidad puede ponerse en duda ante el hecho de que representar todas las reglas con motores de reglas obliga a utilizar este modo para algunas reglas que se implementarían de forma más simple o mejor fuera de estos [22]. Es común que se identifiquen las reglas de negocio implementadas como mecanismos de bases de datos con las restricciones. Date considera estas restricciones como expresiones formales de alguna regla de negocio [74].

Una restricción de integridad es una expresión booleana que tiene que ver con algún requerimiento de una base de datos y debe ser evaluada siempre a verdadero.

Las restricciones deben ser formalmente declaradas en el Sistema de Gestión de Bases de Datos (SGBD), y el SGBD debe hacerlas cumplir, la declaración de estas se realiza usando el lenguaje de la base de datos, hacerlas cumplir es asunto del monitoreo del SGBD.

Date [71] reconoce que en el modelo relacional las restricciones de integridad pueden ser de atributo, de tupla y de bases de datos.

La mayoría de los SGBD cuentan con diversos mecanismos para hacer cumplir las restricciones de integridad: llaves primarias y llaves extranjeras, las cláusulas CHECK, los disparadores, procedimientos almacenados, vistas. Se pueden implementar algunas reglas de negocio como restricciones de integridad, aquellas que restringen datos y sus relaciones.

1.4.1. ***Restricciones CHECK***

Las cláusulas CHECK pueden estar asociadas a una columna limitando los valores que esta puede aceptar o asociada a la tabla limitando los valores que pueden tomar los valores de varios atributos de una tupla [32, 72].

Ventajas

La incorporación de las restricciones en la base de datos evitará cualquier acción que incumpla la regla: cuando el elemento se crea, y si se intenta actualizar. Está relacionado con los límites de los valores de los datos para preservar la integridad de la base de datos. [75], [87].

Desventajas

La principal desventaja de estas restricciones simples es que no se puede utilizar para condiciones más complicadas que implican varias tablas en un sistema de base de datos relacional.

1.4.2. ***Disparadores***

Los disparadores (triggers en inglés) son procedimientos almacenados precompilados en las bases de datos y son invocados automáticamente siempre que algún evento de inserción, actualización o borrado ocurra.

A pesar de que el disparador sucede por un evento en una tabla específica, no se limita al control de las características de esa tabla solamente, sino que se puede utilizar para aplicar restricciones dentro de una tabla, a través de tablas, a través de bases de datos, e incluso a través de servidores. Amplían la lógica de comprobación de integridad, valores predeterminados y reglas del estándar SQL. En [73] se coincide en priorizar el uso de cláusulas Check y los valores predeterminados siempre que estos aporten toda la funcionalidad necesaria. Pueden traer daños si se usan por usuarios inexpertos [87].

Ventajas

Pueden incluir instrucciones SQL complejas. También son útiles para exigir la integridad referencial, que conserva las relaciones definidas entre tablas [75].

Desventajas

- El alto costo de la energía intelectual requerida para crearlos.
- Los factores desencadenantes son más lentos que las restricciones simples.

1.4.3. ***Los procedimientos almacenados***

Es una colección precompilada de instrucciones SQL e instrucciones de control de flujo opcionales, almacenadas bajo un solo nombre y procesadas como una unidad; se guardan en una base de datos, permitiendo ser ejecutados desde una aplicación [88]. Permiten la ejecución de una serie de instrucciones SQL como una unidad. Se ejecuta con mayor rapidez que las instrucciones SQL individuales [75, 87]; un procedimiento almacenado puede invocar a otro procedimiento almacenado con lo que se puede simplificar una serie de instrucciones complejas [88].

1.4.4. ***Vistas***

Una vista es una tabla virtual cuyo contenido está definido por una consulta. Suele utilizarse para centrar, simplificar y personalizar la percepción de la base de datos para cada usuario [73].

Permite a los usuarios centrarse en datos de su interés y en tareas específicas de las que son responsables. Los datos innecesarios pueden quedar fuera de la vista; de ese modo, también es mayor su seguridad, puesto que los usuarios solo pueden ver los definidos en la vista y no los que hay en la tabla subyacente [75].

1.5. Tendencias en el desarrollo de herramientas para gestión de reglas de negocio

Se evidencia una fuerte tendencia de implementar las reglas automáticamente usando los motores de reglas o los SGRN como una buena solución basada en BRA como ILOG Business Rules [2] Blaze Advisor (Fair Isaac Corporation), CleverPathAion Business Rule (Computer Associates International, Inc.), Visual Rule Studio (RuleMachines Corporation), *OpenRules* [3], *JBoss Rules* [4], *RuleXPress*.

Para manejar las reglas de negocio, teniendo en cuenta cómo se representan, implementan y ejecutan, estas herramientas se pueden clasificar como [69, 89, 90]:

- Herramientas independientes de los gestores de bases de datos: las reglas se implementan en la BD, pero los recursos son generados automáticamente y manejadas por herramientas que no son gestores de bases de datos.
- Herramientas basadas en servicios: las reglas son creadas por herramientas de desarrollo que constituyen una capa intermedia de servicios de aplicación y residen en el servidor de aplicaciones.
- Sistemas basados en reglas: Se captura la lógica del negocio a un alto nivel y las reglas asociadas, se usan métodos orientados a la lógica. En tiempo de ejecución se usan motores especiales para procesar las reglas y generar respuestas apropiadas.

Una de las actividades más importantes en el desarrollo de SI basado en reglas, es la consideración de diferentes tecnologías y herramientas disponibles para apoyar la implementación de reglas, incluida la generación de código [69, 87].

La tecnología a utilizar depende de varios factores, pero particularmente en el tipo de sistema en desarrollo. Por ejemplo, en una aplicación típica basada en la aplicación del conocimiento, las reglas se capturan y se almacenan en una base de reglas y se ejecutan por un motor de reglas. Sin embargo, en un sistema típico de flujo de trabajo, las reglas de negocio se integrarán en la

[2] http://www.ilog.com/products/jrules/whatsnew/

[3] Business and Technical Specialists Working Together On a Rule Project. Part 1 - Creating a Basic Rule Project http://openrules.com/docs/ClinicalGuidelines.Part1.htm

[4] JBoss Enterprise BRMS Platform 5 BRMS UserguideForJBoss Rules developers, rule authors, and business analysts. Edition 5.1.0

definición del flujo de trabajo y será utilizado un sistema de flujo de trabajo para ejecutar el mismo.

Entre los componentes fundamentales para trabajar con reglas de negocio está el repositorio de reglas para almacenar las declaraciones de las reglas y lograr algún grado de independencia de las reglas con los SI. Existen en la literatura diferentes propuestas de estructura de repositorios de reglas [91, 92].

Otro de los desafíos en la implementación de reglas de negocio es proporcionar una herramienta de apoyo para la generación de código de programa (para la ejecución de las reglas de negocio) a partir de las especificaciones de reglas de negocio. Las herramientas más comercializadas que están disponibles hoy día y ofrecen este tipo de funcionalidad son para determinadas tecnologías. Si se utiliza una herramienta de motor de reglas de inferencia, el código de programa sólo puede ser generado para el uso de ese motor de reglas. Además, dado que las especificaciones deben ser rigurosas a interpretarlas por una computadora, las herramientas a menudo sólo admiten un número limitado de categorías de reglas.

Por ejemplo, Athenas [38] presenta un motor de ejecución de reglas de negocio sobre el sistema de gestión de Oracle, para implementar automáticamente un tipo de regla, las reconocidas como reglas de restricción.

En ese trabajo se tiene en cuenta que la implementación de las reglas en la capa de datos es una manera de trabajar con aplicaciones que con mínimos cambios pueden referirse a bases de datos que generalmente no cambian [38].

1.6. Repositorio de reglas

El repositorio de reglas es uno de los componentes fundamentales para trabajar con reglas de negocio y lograr independencia de las reglas y los SI. El repositorio almacena la declaración de las reglas y aspectos que permiten garantizar su control y mantenimiento. Existen en la literatura numerosos esquemas de repositorios de reglas [91, 92].

En [91] se hace una propuesta que corresponde con la de un repositorio de reglas para todas las reglas con las que se trabaja, independientemente de la implementación de las mismas. Cada regla de negocio debe describirse en el repositorio por un conjunto de atributos, entre los más importantes están: el identificador de la regla, número de la versión de la regla correspondiente, la fecha de creación o del último cambio, el tipo de regla, el estado, la sentencia de la regla en sí. Se destaca que las reglas pueden ser aplicadas por determinado período de tiempo.

Una característica de esta propuesta [91] es que el repositorio muestra trazabilidad desde los modelos de negocio que soportan su definición primaria a sus realizaciones en el sistema de información.

1.7. Influencia de las modificaciones de las reglas en los SI

Los cambios en el entorno empresarial de una organización pueden ser impulsados por decisiones internas de la organización o por fuerzas externas. Estos cambios con frecuencia requieren propagarse en los sistemas nuevos o modificar los existentes [93]. Para garantizar la consistencia del negocio y los SI asociados, se debe controlar el conjunto de acciones permitidas para estas modificaciones.

Según [93], además de las actividades del modelado de la empresa y el desarrollo de un SI, el escenario manejador de reglas de negocio prescribe tareas adicionales que se ocupan de los cambios de las mismas. Estas actividades se pueden realizar a nivel de empresa o del SI, ellos incluyen: control de cambios, control de versiones y control de impacto. Las reglas de negocio rara vez son independientes, lo que significa que un cambio en una regla en particular puede causar cambios en otras reglas.

Una versión de un objeto dado se crea a partir de las modificaciones hechas en el tiempo a las versiones previas, partiendo de una versión inicial. Hacer versiones incluye que no se pierda información valiosa. Para ello es preciso definir reglas de integridad y mecanismos que controlen la evolución [92, 94, 95]. Aunque en determinados sistemas se opta porque al modificar la regla, su versión anterior no tendrá validez. Algunos usan una u otra alternativa [38].

En [92, 95, 96] se propone resolver el problema de versiones de reglas de una manera declarativa, partiendo de que todas las reglas están sujetas a ser versionadas y como condición extra se tiene que pueden estar activas en un período de tiempo determinado, se utiliza un atributo de marca de tiempo (Mark date) con la fecha de comienzo y fecha fin en que la regla estuvo activa.

Se ofrecen un conjunto de enunciados [92] que describen el problema de versiones de reglas:

- Las reglas son sentencias declarativas que son usadas en una o más tareas para determinar un cierto valor para un atributo.
- Algunas reglas solo pueden ser aplicadas en un período de tiempo determinado.
- Las reglas aplicables sobre cierta fecha deben ser consistentes.

- Un caso del pasado puede ser retomado aplicando las reglas que estaban activas en ese período.
- Las fechas que se usan para recuperar las reglas correctas pueden ser diferentes de acuerdo con: la tarea que se va a llevar a cabo, el caso (situación) sobre el cual las reglas se aplicarán, o el evento que produjo la activación de las reglas. En [95] las reglas se agrupan en conjuntos, de modo que las reglas de un conjunto pueden ser activadas en el mismo período de tiempo. Se utilizan algoritmos de encadenamiento hacia adelante y hacia atrás para los motores de reglas que tienen un mecanismo de inferencia. La desventaja de esta solución está en el mantenimiento del conjunto de reglas; el cual contendrá gran cantidad de redundancias pues hay poca superposición en el período de aplicabilidad de las mismas.

En [96] se discute una estrategia para tratar con versiones de reglas, la cual trabaja si hay lotes de diferentes versiones. Para versionar reglas complejas recomienda ver las reglas como objetos con atributos (meta información) y crear un objeto para cada regla. Esto permite especificar una información adicional de estas en su modelo de objetos.

1.8. Conclusiones parciales

Los investigadores de las reglas de negocio tienden a categorizar las reglas para facilitar el tratamiento de las mismas desde su origen por los expertos del negocio, hasta las implementaciones en los SI.

Se observa que existen diversos autores que entre sus categorías involucran algunas reglas que se pueden asociar a los datos del negocio, los cuales se almacenan generalmente en bases de datos relacionales. Estas reglas pudieran ser, en última instancia, comprobadas o ejecutadas ante operaciones sobre los datos.

Se evidencia la diversidad de lenguajes existentes para formalizar la escritura de las reglas de negocio, cada uno de ellos responden a un nivel de abstracción que puede corresponder a los niveles: informal o natural estructurado; técnico y formal (ya cerca de la implementación), algunos buscan la interoperabilidad de las reglas en la Web; se desarrollan lenguajes que responden al trabajo con motores de reglas específicos; sin embargo, el costo de aprenderlos y usarlos es alto. En la literatura especializada no se observa ningún formalismo para expresar las reglas en función de los términos del negocio reflejados en las tablas y atributos de las bases de

datos relacionales, ni que se pueda utilizar tal y como se define, ni sea de fácil trasformación; con OCL se pueden representar solo reglas de restricción.

Con la tendencia de gestionar las reglas de negocio con diferentes tipos de herramientas que permiten su generación automática, se vislumbra la posibilidad de no forzar la implementación de todas las reglas en motores de reglas de inferencia; pues aunque es una manera promocionada y muy utilizada en los últimos tiempos, tiene algunas desventajas de importancia a tener en cuenta.

La adopción de uno de los motores de reglas de inferencia, comercializados actualmente, requiere una gran inversión de tiempo así como otras desventajas estudiadas, además obliga en algunos casos a complejizar la implementación de reglas que pudieran programarse de manera más sencilla. Pudiera analizarse la utilización de los recursos SQL de las bases de datos si las reglas están asociadas a los datos.

Para la generación de las implementaciones de reglas de manera automática, se deben tener en cuenta estrategias para controlar las posibles modificaciones de las mismas como el control de versiones y el historial de reglas. Para aquellas cuya implementación ha sido automáticamente insertada en bases de datos, este aspecto también debe tratarse con cuidado.

2. PROPUESTA DE CATEGORÍAS DESDE LA PERSPECTIVA DE LOS DATOS

En este capítulo se proponen un conjunto de categorías de reglas de negocio, que han de cumplirse sobre los datos; un lenguaje de nivel técnico de patrones para expresarlas nombrado LPT; así como las consideraciones y los pasos para traducir las reglas de LPT a lenguaje SQL en forma de mecanismos de bases de datos. Se expone, por último, una manera de tratar las modificaciones de las reglas y su propagación sobre los datos.

2.1. Categorías de reglas de negocio desde la perspectiva de los datos

Como se explicó en el capítulo 1 dentro del conjunto de las reglas de negocio hay un grupo de ellas que se asocian al comportamiento de los datos del negocio. No se ha encontrado una clasificación que agrupe a estas reglas con esta perspectiva para estudiarlas, implementarlas, etc.

Una regla de negocio desde la perspectiva de los datos es aquella que se define sobre los términos del negocio, entidades y atributos del mismo; ante la adición, modificación o eliminación de datos debe ser evaluada. Se proponen cuatro tipos de reglas, organizadas en sus patrones correspondientes [97, 98]:

Restricción: Hace cumplir políticas prohibitorias en el sistema del negocio.

Cómputo: Calcula un valor determinado en el negocio y en ocasiones hace que un atributo adquiera el valor declarado en la regla.

Notificación: Alerta sobre alguna situación de incumplimiento, en tiempo real, pero no impide que la situación suceda.

Clasificación: Define un concepto, que no se refleja en la estructura del negocio.

Los convenios utilizados para precisar elementos variables de los patrones se corresponden con la forma de Naur Backus (FNB).

Los significados de los elementos del patrón aparecen en la Tabla 2.1.

Tabla 2.1 Elementos para expresar las reglas de negocio "desde la perspectiva de los datos".

Elemento	Significado
<determinante>	Es el determinante para cada sujeto, por ejemplo: Una, Uno, El, La, Cada, Todos. Según el mejor sentido en la redacción de la regla.
<sujeto>	Es un término u objeto del negocio, tipo de entidad.
<hechos>	Son hechos relativos al estado o comportamiento del negocio, incluyendo o no al sujeto.
<características>	Describe las características del sujeto en el negocio, tanto internas como relacionadas con otras entidades. Pueden incluir hechos con el fin de caracterizar al sujeto.
<resultado>	Es cualquier valor numérico que tiene significado en el negocio y resulta de evaluar una expresión matemática.
<expresión matemática>	Es una expresión matemática que se define sobre combinaciones de términos del negocio junto con funciones y operadores disponibles, para obtener un resultado.
<atributo>	Es un atributo de una entidad del negocio. Un resultado puede destinarse para un atributo.
<clasificación>	Definición de un término del negocio, que se refiere a un subconjunto de instancias de un tipo de entidad del negocio que cumple determinadas características.
<mensaje>	Mensaje de información sobre estado de alerta del negocio.

Fuente: Elaboración propia, basado en [22].

Los <hechos>, las <características> y los elementos que operan en la expresión se escriben en lenguaje natural.

A continuación se especifican los patrones definidos para cada categoría [97-100]. En los ejemplos que siguen se utilizará una parte de las reglas de negocio encontradas en un sistema de control de trasplantes renales y en un sistema de control de estudiantes.

2.1.1. ***Patrón para Reglas de Restricción***

Este patrón es útil para satisfacer requisitos del negocio, solo con reglas de este tipo puede hacerse cumplir políticas prohibitorias en el sistema del negocio [99, 101-108]. Tiene la siguiente estructura:

```
<determinante> <sujeto> (no puede tener <características>) |
                (puede tener <características> solo si <hechos>)
```

Esta categoría de regla describe restricciones para el <sujeto> de la regla. Al <sujeto> se le restringen ciertas <características> a excepción de determinados <hechos>. La definición de los <hechos> puede o no referirse al sujeto.

Ejemplo RN# 1: Un cirujano no puede tener (atender) más donantes vivos que la sexta parte de todos los donantes vivos.

Aquí el sujeto de la regla es "cirujano" y características se refiere a la atención a menos de la sexta parte de todos los donantes vivos. Esta regla se considera "desde la perspectiva de los datos" pues debe evaluarse cada vez que se asigne a un cirujano la atención de un donante vivo.

2.1.2. ***Patrón para Reglas de Cómputo***

Su objetivo es calcular un valor determinado en el negocio, asociado o no a un <sujeto> y su resultado es numérico. Ayuda a definir nuevos datos [97, 99, 106-108]. Tiene la siguiente estructura:

```
<determinante> <resultado> [en <sujeto> [para <atributo>]]
                        es calculado como <expresión matemática>
```

Cuando no se especifica <sujeto> se está aplicando la <expresión matemática> sobre un conjunto de instancias, o en caso que se especifique se aplica para una en particular. El <resultado> puede ser o no un atributo del <sujeto>.

Para los ejemplos véase el diagrama conceptual de la figura 2.1

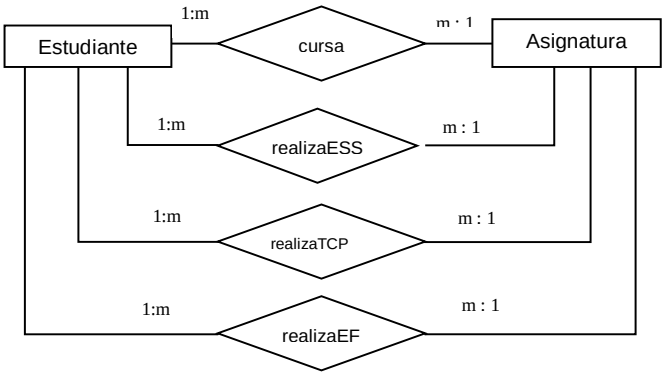

Figura 2.1 Fragmento conceptual de la BD "Control de Evaluaciones"

Aunque el patrón es general, para este tipo de regla se especifican tres variantes, con una estructura específica cada una:

Variante 1 (V1): `<determinante><resultado> es calculado como <expresión matemática>`

El resultado no es un atributo de la BD y las expresiones de la <expresión matemática> involucran a todas las instancias de las entidades.

Ejemplo RN#2: La matrícula del grupo es calculada como la cantidad de estudiantes.

Variante 2 (V2): `<determinante> <resultado> en <sujeto> es calculado como <expresión matemática>`

Calcula un resultado que se desea obtener para una instancia específica de la entidad <sujeto>. El resultado no se almacena en un atributo de la BD.

Es útil para el cálculo de un resultado que se desea obtener para una instancia. Por ejemplo: el promedio de un estudiante, el salario de un trabajador, etc.

Ejemplo RN#3: El promedio en estudiante es calculado como la suma del promedio de los trabajos de control, promedio de realizaESS (asignatura cursada por el estudiante) y el máximo de las notas de los exámenes finales.

Variante 3 (V3): `<determinante><resultado> en <sujeto> para <atributo> es calculado como <expresión matemática>`

El resultado se guarda en un atributo de la BD, para cada instancia del <sujeto>.

Ejemplo RN#4: La nota de trabajo de control parcial en Estudiante para TCP_Total es calculada como el promedio de los trabajos de control parcial de cada asignatura.

2.1.3. ***Patrón para la Regla de Notificación***

La utilidad de este tipo de regla está en alertar sobre alguna situación de incumplimiento, en tiempo real, pero no impide que la situación suceda. En [40] se afirma que si se piensa solo en reglas de negocio como restricciones complejas y rápidas se pierde una parte importante del aspecto global de las mismas. Este patrón puede ayudar a la toma de decisiones en el negocio, son similares las restricciones suaves de [15]. Tiene la siguiente estructura [97, 99, 106, 108] :

```
Notificar <mensaje> si <hecho>
```

Ejemplo RN#5: Notificar "Alerta de donantes con riesgo" si la cantidad de donantes potenciales con resultado de Evolución Positivo es mayor que 80.

2.1.4. ***Patrón para la regla de Clasificación***

El patrón de clasificación puede ser visto como una regla de definición que no se refleja en el diseño estructural de la base de datos. Estas reglas reflejan un conocimiento básico del negocio contribuyendo claramente al significado de conceptos, centrándose en la esencia del mismo. Tiene la siguiente estructura [97, 99, 106, 108]:

```
<determinante> <sujeto> [no] es definido como <clasificación>
                [( si | a menos que )<característica>]
```

Este patrón comprueba si determinado <sujeto> puede o no ser clasificado como <clasificación>. Se crea una <clasificación>, que pertenece al vocabulario del negocio y puede ser utilizada por otras reglas que estarán implicadas directamente en operaciones sobre los datos.

Ejemplo RN#6: La evolución de un paciente es definido(a) como evolución satisfactoria si su temperatura es menor de 37ºC.

Este grupo de patrones que se ha especificado presenta las reglas de negocio denominadas en el presente trabajo "desde la perspectiva de los datos", o sea, aquellas que tienen relación directa con los datos de una base de datos, independiente del modelo que se utilice para su creación. Además se ha brindado una manera de escribirlas en el nivel informal.

Hay un conjunto de otras reglas de negocio que no abarcan estos patrones y que no son objeto de estudio en el presente trabajo.

2.2. LPT: Lenguaje de patrones técnico

La idea fundamental del presente trabajo es expresar las reglas de negocio desde la perspectiva de los datos e implementarlas automáticamente en la base de datos; esto, como se analizó en el capítulo 1, puede proporcionar flexibilidad a los SI implementados y disminuir el costo de implementación.

Se pretende expresar las reglas de negocio sobre las tablas [5] de una base de datos ya creada y para ello se propone un lenguaje de nivel técnico que permitirá la expresión de todas las categorías de reglas de negocio desde la perspectiva de los datos. A este lenguaje se le ha nombrado LPT (Lenguaje de Patrones Técnico) [97, 103, 109, 110].

Las especificaciones que se hacen en LPT para nombrar las tablas (relaciones) y los atributos de la base de datos coinciden con los términos utilizados en el modelo de hechos relacionado con el negocio [111], implícito en el esquema relacional, especificado en el catálogo de la base de datos. Se propone la navegación a través de las tablas que constituyen términos del negocio, de los que se necesita guardar información. LPT utiliza la notación punto para establecer el acceso a los atributos de tablas y posibilitar la navegación [99, 103], permitiendo la expresión de todas las categorías de reglas de negocio desde la perspectiva de los datos.

Esto no es un problema, porque como se plantea en [111], las reglas de negocio se construyen tomando como base un vocabulario que está formado por las palabras y frases establecidas por la comunidad de usuarios de un negocio específico. Aquí la modelación de las mismas supone un uso controlado y previo de dicho vocabulario.

Se trabaja con el esquema relacional, especificado en el catálogo de la base de datos, se propone la navegación a través de las tablas que representan términos del negocio de los que se necesita guardar información.

La estructura del LPT brinda tres facilidades para un especialista técnico del negocio: la compresión relativamente fácil, la simple transformación desde los patrones de reglas en lenguaje natural, así como la escritura simple usando notación punto.

Este lenguaje busca un equilibrio entre las funcionalidades similares al OCL y la necesidad de conservar la simplicidad deseada, que cuenta solo variantes indispensables implícitas en su

[5] Aunque la estructura básica del modelo relacional es la relación en el presente trabajo se utilizarán indistintamente los términos de relación y tabla, por ser este último muy difundido en el mundo de los gestores de datos

definición y basado en patrones de reglas desde la perspectiva de los datos. Se utiliza la notación punto como estilo, para establecer el medio de acceso a los atributos de tablas y posibilitar la navegación.

2.2.1. ***Sintaxis y semántica del LPT***

La especificación completa de la sintaxis y semántica de LPT, así como sus operadores se describen en el Anexo 1 "Sintaxis del LPT" y "Semántica del LPT", respectivamente.

A continuación se mostrarán algunos elementos esenciales de las mismas, solo con el objetivo de mostrar las funcionalidades principales [110, 112].

Acceso simple a un atributo: Tabla1[.Atributo]. Referencia a los valores que toma el Atributo en la Tabla1. Si no aparece *.Atributo*, se referencian implícitamente los valores de los atributos identificadores.

Camino de navegación entre tablas: Tabla1.Tabla2. (…) .TablaN [. Atributo]

A Tabla1.Tabla2.Tabla3…TablaN se le llama camino de navegación, porque indica una manera de ir de la Tabla1 a la TablaN.

Cuando el camino incluye varias tablas se devuelven uno o más valores del atributo en dependencia de las interrelaciones existentes entre las tablas.

Los elementos individuales se obtienen cuando el camino de navegación tiene varias tablas con interrelaciones 1:1 o solamente una tabla, útiles para el acceso a atributos.

Los elementos múltiples se obtienen como resultado de un camino de navegación con varias tablas y al menos una interrelación uno-muchos.

En esta notación se emplea como palabra reservada *sujeto*, para referenciar la tabla del <sujeto> de la regla. Esta forma es similar al uso del *this* de los diferentes lenguajes de programación [113]. Sobre el diagrama de la figura 2.2 para expresar: "El cirujano del donante potencial cuya evolución …" se escribe (Donante_Potencial.Cirujano…), la referencia se realiza a un único cirujano para cada instancia de Donante_Potencial. (elementos individuales). Como se observa el orden de la notación es muy importante.

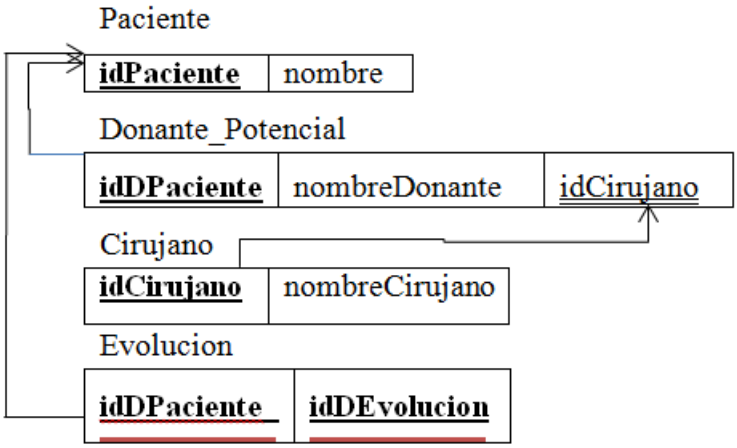

Figura 2.2 Parte del esquema relacional para trasplante renal

Las Evoluciones de Pacientes…" se expresa (Paciente.Evolucion) y se referencian varias evoluciones relacionadas con un único Paciente.

También se puede restringir el resultado a partir del valor de algún atributo perteneciente a cualquier tabla que pertenezca al camino, de la forma

Tabla**(**[Atributo OperadordeComparacion] valor**)**

Así *"el nombre del Paciente con carnet de identidad 87032215468"*, se escribe en LPT como: *Paciente (CI='87032215468').nombre*.

Operadores aritméticos, lógicos, de comparación y de conjunto del LPT.

Los operadores definidos para el LPT son los operadores básicos para construir sentencias complejas y se muestran como parte de la sintaxis y semántica del lenguaje LPT, véase Anexo 1.

Los operadores de comparación (>|<|>=|<=|=|<>) pueden utilizarse indistintamente para comparar elementos individuales y múltiples, las combinaciones posibles de estos son:

(ElementoIndividual1)operador_de_comparación (ElementoIndividual2)

donde se compara el ElementoIndividual2 con el ElementoIndividual1

(ElementoMúltiple) operador_de_comparación (ElementoIndividual) *o*

(ElementoIndividual) operador_de_comparación (ElementoMúltiple).

en estos casos el ElementoIndividual se está comparando con todos los elementos del ElementoMúltiple. La condición es verdadera si se cumple que el ElementoIndividual cumple la relación con todos los elementos del ElementoMúltiple.

(ElementoMúltiple1) operador_de_comparación (ElementoMúltiple2)

En este caso los elementos ElementoMúltiple1 están comparándose con los elementos de la segunda colección (ElementoMúltiple2). Esto significa que cada elemento de la colección ElementoMúltiple1 debe cumplir la comparación con todos los elementos de la colección ElementoMúltiple2.

Para comparar dos elementos múltiples en igualdad (=), se debe demostrar que una colección está contenida en otra y viceversa.

Para comparar dos elementos múltiples en desigualdad, se debe demostrar que una colección no está contenida en otra y viceversa.

A continuación se muestran algunos ejemplos:

Ejemplo 1:

sizeof(*Pacientes.ExamenesFísicos.Resultado = 'Dengue'*) > 800

(la cantidad de pacientes que tienen exámenes físico con resultado Dengue es mayor que 800)

Ejemplo 2

Paciente.Evoluciones.Temperatura> 38

equivalente a (38< Paciente.Evoluciones.Temperatura)

(la temperatura de todas las evoluciones de los pacientes es mayor que 38)

Ejemplo 3:

Paciente.Evolucion.idEvolucion>Donante_Potencial.Cirujano.idCirujano

(el id de cada Evolución de un paciente es mayor que el máximo valor del idCirujano de todos los donantes potenciales).

Generalmente no es común tener restricciones entre identificadores de entidades del negocio; es solo un ejemplo a cumplir sobre la figura 2.2.

2.2.2. Ejemplos de Reglas de Negocio expresadas en LPT

A continuación se describen los ejemplos de reglas presentados en el epígrafe 2.1 (en lenguaje natural con patrones o natural estructurado) escritos en el lenguaje técnico LPT.

Patrón en LPT para Reglas de Restricción:

RN# 1: Un cirujano no puede tener (atender) más donantes vivos que la sexta parte de todos los donantes vivos.

Expresión en LPT:

RN#1: Un cirujano no puede tener sujeto.Donantesvivos> (sujeto.Donantesvivos) /6

Patrón en LPT para Reglas de Cálculo.

Como se conoce de este tipo de regla se tienen tres variantes:

Variante 1 (V1): La matrícula del grupo es calculado(a) como la cantidad de estudiantes. Expresión en LPT:

RN#2: matrícula_del_grupo es calculado como sizeof(estudiante.idEstudiante)

Variante 1 (V2): RN#3: El promedio en estudiante es calculado como la suma del promedio de los trabajos de control, el promedio de ES y el máximo de las notas de los exámenes finales.

Expresión en LPT:

RN#3: Promedio en estudiante es calculado como

avg(sujeto.cursa.promedioTCP)+avg(sujeto.cursa.promedioES)+ max(sujeto.EF.nota)

Variante 3 (V3): RN#4: La nota de trabajo de control parcial en Estudiante para TCP_Total es calculado como el promedio de los trabajos de control parcial de cada asignatura.

Expresión en LPT:

RN#4: nota_TCP en Estudiante para TCP_Total es calculado como

avg(sujeto.asignatura.tcp.nota)

Patrón en LPT para la Regla de Notificación

RN#5: Notificar "Alerta de donantes con riesgo" si la cantidad de donantes potenciales cuyo resultado de Evolución en Positivo es mayor que 80.

Expresión en LPT:

RN#5: Notificar "Alerta de donantes en riesgo" si

sizeof(sujeto.evolucion.resultado="Positivo")>80

Esta regla no se infringe, pues su función es calcular y brindar una respuesta.

Patrón en LPT para la Regla de Clasificación

RN#6: La evolución de un paciente es definido(a) como evolución satisfactoria si su temperatura es menor de 37ºC.

Expresión en LPT:

RN#6: La evolución es definida como evolución_satisfactoria si sujeto.temperatura<37

2.3. Interdependencia de las Reglas de Negocio para las categorías desde una perspectiva de los datos

Entre estas categorías de reglas de negocio pueden existir interdependencias entre las reglas.

Supongamos los siguientes ejemplos de reglas de negocio:

RN#1: Síntomas_Históricos_Negativos de un Paciente es calculado como la cantidad de Evoluciones que tienen temperaturas mayores de 38ºC.

RN#2: Un Paciente es definido como Riesgo_Potencial si sus Síntomas_Históricos_ Negativos son mayores que 80 o el promedio de las medidas de las tensiones arteriales registradas en las evoluciones es mayor que 160.

RN#3: Un Riesgo_Potencial no puede tener menos de 10 Exámenes ni ser un Donante Potencial.

Obsérvese que la regla RN3 utiliza en el <sujeto> una *clasificación de un paciente (Riesgo Potencial)*, por tanto depende de la regla que define *Riesgo_Potencial,* RN#2. La regla RN#2 también maneja el término Síntomas_Históricos_Negativos que se definen en la regla RN#1 que define un cálculo (Síntomas Históricos Negativos) y es independiente.

Para los patrones definidos en esta investigación se muestran sus posibles interdependencias en la tabla 2.2.

Tabla 2.2 Interdependencias entre reglas desde la perspectiva de los datos.

Tipo de Regla	*Posibles Reglas a depender*
Clasificación	Clasificación Cómputo
Cómputo	Cómputo Clasificación
Notificación	Clasificación Cómputo
Restricción	Clasificación Cómputo

Fuente: Elaboración propia.

Una posible interdependencia entre las reglas se puede mostrar mediante un grafo dirigido:

En la figura 2.3 se muestra como la Regla R1 depende de las reglas R2 y R3, y R2 depende de R4.

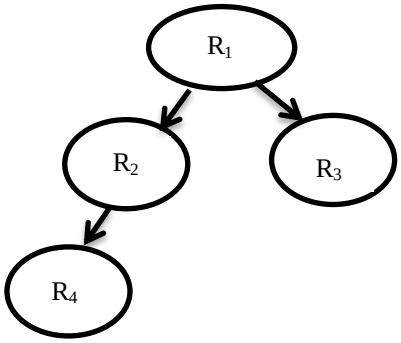

Figura 2.3 Interdependencias entre Reglas de Negocio.

La interdependencia de reglas pudiese generar ciclos. Los ciclos deben ser evitados, como señala también [114], pero el tratamiento de ciclos no es objeto de estudio en esta investigación.

2.4. Traducción de los caminos de navegación a recursos de bases de datos

Para la ejecución de las reglas que pertenecen a las categorías "desde la perspectiva de los datos" se propone utilizar código SQL.

Para lograr sentencias SQL que permitan hacer cumplir las reglas se definen algunos elementos necesarios para poder "traducir" las reglas escritas en LPT a SQL [105, 115].

Uno de los primeros aspectos que se debe considerar es, cómo lograr expresar los caminos representados por la notación punto [110].

Se tiene que un esquema relacional según [116] puede ser representado mediante un grafo, donde los nodos representan relaciones del diagrama relacional y las aristas son interrelaciones que tienen lugar en el esquema, que se expresan a partir de las llaves foráneas y están orientadas según el sentido de estas.

Si E1, E2 y E3 son relaciones que se identifican con términos del negocio y están relacionados de la manera que muestra la figura 2.4, significa que E2 contiene un atributo (o conjunto de atributos) que referencia a E1 y E3 tiene un atributo (o conjunto de atributos) que referencia a E2.

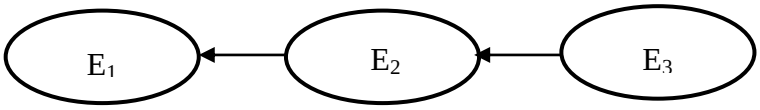

Figura 2.4 Dígrafo que representa un camino de navegación para una regla de negocio R1

Entonces si una regla de negocio escrita en LPT contiene el camino de navegación $E_1.E_2.E_3$.Atributo, se puede decir que este camino de navegación corresponde exactamente con un camino en el grafo que representa al esquema relacional.

Al camino de navegación correspondiente al elemento de la regla escrita en LPT se le denominará *camino de navegación explícito*, mientras que a un camino que se corresponde con el grafo se le denominará *camino de navegación completo.* En el caso de la figura 2.4 coincide el camino de navegación explícito con el camino de navegación completo [117].

Ante un evento de inserción, actualización o eliminación en algunas de estas tablas es necesario conocer en qué tablas del camino de navegación se analiza el cumplimiento de la regla.

El camino de navegación explícito y el completo no siempre coinciden. Supóngase el caso del diagrama relacional de la figura 2.5, y una regla que especifica un camino de navegación $E_1.E_3$.atributo, en el que E_1 y E_3 son términos identificados del negocio mientras E_2 no lo es.

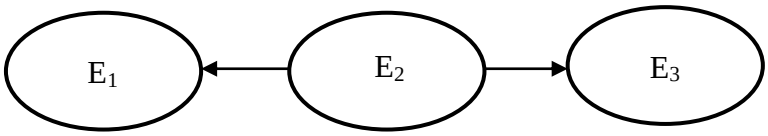

Figura 2.5 Dígrafo de las características para una regla de negocio R2

La tabla E_2 existe por una necesidad del diseño de la base de datos (un caso típico es la solución que se da para la interrelaciones muchos-muchos). Por lo tanto para ir de E_1 a E_3 hay que pasar necesariamente por E_2, por lo que el camino completo sería $E_1E_2E_3$. A esta tabla que completa el camino de navegación entre las tablas con que se expresa la regla se le denomina *tabla de resolución* [117].

Una *tabla de resolución* se caracteriza por las siguientes características:

- Sus llaves foráneas son las llaves primarias de las tablas que corresponden a las tablas de términos del negocio que intervienen en la regla.
- Sus llaves foráneas en cuestión están contenidas en la llave primaria de la propia tabla (tabla de resolución).

Se propone que *cada camino de navegación completo* de cada regla se transforme en una expresión válida SQL en un entorno relacional. A la sentencia SELECT que se conforma para cada regla se le denomina *consulta base de la regla* y consiste básicamente en un acople entre las tablas que componen el camino de navegación completo.

A partir del camino de navegación completo se identificarán cuáles tablas de este camino deberá analizarse el cumplimiento de la regla. No es necesario chequear todas las tablas del camino de navegación completo, *sino* aquellas que tienen aristas salientes, según se analiza, son tablas de las

que al menos hay una arista saliente (cant Nodo (aristas saliente)>=1), a una tabla así se le llama *tabla común*. Una *tabla de resolución* constituye un *tipo de la tabla común*.

Sobre las *tablas comunes* debe chequearse el cumplimiento de la regla cuando ocurre un evento sobre una tabla del camino de navegación en que ella está involucrada por la regla que se analiza.

En el caso que <características>, <hechos> y <expresión matemática> involucren solamente a atributos de la tabla sujeto, se evaluará la regla sobre esta tabla.

De la manera descrita anteriormente se conforman los caminos de navegación completos para los elementos de las reglas: <características>, <hechos> y <expresión matemática> escritos en LPT como caminos de navegación explícitos entre tablas.

Creación de las Listas de Eventos

Se considera un evento, a una operación de inserción, modificación o eliminación sobre los datos. La ejecución de estas operaciones hace que sea necesario evaluar las reglas de negocio que se han clasificado "desde la perspectiva de los datos".

Una lista de eventos revela sobre cuáles tablas comunes asociadas a las <características>, a los <hechos>, <expresión_matemática> de una regla expresada en LPT, debe evaluarse cada regla, para un tipo de evento sobre la tabla. Para cada regla se crea una lista por tipo de evento. Así, para la Regla j, con el camino de navegación completo cuya lista ordenada de n tablas está representa como $\{E_1, E_2,\ldots, E_n\}$, las listas de eventos correspondientes son:

LE Inserción $\{V_1, V_2,\ldots, V_n\}$

LE Modificación $\{V_1, V_2,\ldots, V_n\}$

LE Eliminación $\{V_1, V_2,\ldots, V_n\}$

Donde:

Para el evento de inserción

$$Vi = \begin{cases} 0, \text{ } E_i \text{ tiene al menos una arista entrante} \\ 1, \text{ } E_i \text{ solo tiene aristas salientes} \end{cases}$$

Para los eventos de modificación y eliminación

$$Vi = \begin{cases} 0, \text{ } E_i \text{ no tiene aristas salientes} \\ 1, \text{ } E_i \text{ tiene al menos una arista saliente} \end{cases}$$

Para el ejemplo de la figura 2.4; si se supone que los nodos que corresponden a una Regla j, son E_1, E_2, E_3 entonces a la tabla del sujeto E_1, le corresponde el valor V_1, a la tabla E_2 *el valor* V_2 *y* a la tabla E_3, V_3.

La lista de eventos para inserción quedaría como LE_j (0,1,0). El valor 1 significa que sobre la tabla correspondiente en la posición dos (E_2) de la lista de tablas se chequeará el cumplimiento de la regla. En este caso para la regla analizada y el evento de inserción se chequea en la tabla E_2, pues es el único nodo con todas sus aristas salientes.

Teniendo en cuenta el dígrafo conformado, se propone una manera de proceder para conformar la lista de eventos para cada operación sobre los datos.

En resumen, la evaluación de las reglas se realizará para cada evento en los nodos o tablas del camino de navegación que se indican a continuación:

Eventos de inserción.

- En nodos con todas sus aristas salientes.

Eventos de actualización.

- En el atributo del último nodo del camino de navegación (si posee atributo).
- En nodos con al menos una arista saliente.

Evento de eliminación.

- En los nodos con al menos una arista saliente, excepto el nodo que esté asociado al sujeto de la regla.

Las acciones principales que se han revelado para localizar las tablas y los eventos sobre los datos para evaluar las reglas son [118]:

1. Distinguir los términos del negocio y las tablas de resolución en el camino de navegación.
2. Reconstruir la regla por medio de la inserción de las tablas de resolución, en caso necesario, para conformar los caminos de navegación completos implicados.
3. Construir la lista de eventos (LE).

Estas acciones para localizar las tablas y los eventos sobre los que se evalúan las reglas quedan detalladas en el siguiente algoritmo:

PASO 1. Crear el digrafo G_{CN} que representa el esquema lógico de la base de datos. A partir del catálogo de la base de datos se inserta en el dígrafo un nodo por cada tabla y una arista por cada llave primaria-llave extranjera que se encuentra.

Sea V el conjunto de nodos o vértices del grafo.

PASO 2: Sea la regla R_i, C el conjunto de todos los elementos de un camino de navegación de la regla ordenado en la secuencia que aparece.

Para i=1, |C| -1

Buscar E_i € C en V

Determinar si existe el arco E_i,E_{i+1} en el grafo G_{CN},

Si no existe buscar E_j tal que, existe el arco E_i,E_j y el arco E_j, E_{i+1}.

E_j constituye una tabla de resolución, con la cual se completa el camino de navegación que se analiza formando el camino de navegación completo E_i.E_j. E_{i+1}.

Fin Para

Paso 3: Construir la lista de eventos (LE) para la Regla R_i.

Con el camino de navegación completo cuya lista ordenada de |C| tablas está representada como $\{E_1, E_2, \ldots, E_n\}$ las listas de eventos correspondientes son:

Crear LEi , LEu, LEd vacías,

Para i=1, |C $_{\text{camino navegación completo}}$|

Buscar E_i en V

Si grado de entrada(E_i)=0, añadir 1 a LEi, sino añadir 0

Si grado de salida(E_i)=0 añadir 0 a LEu y LEd, sino añadir 1.

Fin Para

La complejidad temporal de este algoritmo es O ($|v|.|c|^2$)

2.5. Tratamiento de las reglas con interdependencias. Consideraciones generales

Cuando una regla depende de otras, la regla principal o regla dependiente, debe considerar las listas de evento (LE) de las reglas de las cuales depende.

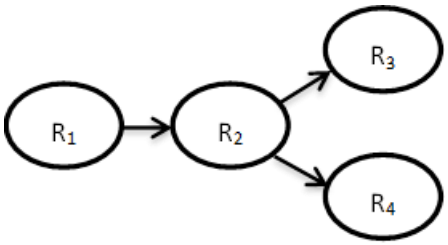

Figura 2.6 Interacción entre reglas

Cuando una regla depende de otras, a la LE correspondiente de la regla principal se le adicionan las listas de eventos (LE) de las reglas que participan. Para una interdependencia de reglas como la mostrada en la figura 2.6, donde la regla R1 depende de R2 y a su vez R2 depende de R3 y R4, la generación y/o activación de la regla R2 con R3 y R4 activadas previamente, y después R1 implica que se ejecute una serie de pasos que se conciben como un algoritmo.

Este algoritmo que expresa la creación de las interdependencias se presenta a continuación:

Se crea un grafo no conexo con las interdependencias de las reglas, cada componente conexa expresa un conjunto de reglas que se relacionan entre sí.

Este grafo se va creando a medida que se analiza cada una de las reglas.

Sea el conjunto R de reglas a analizar. Se tiene el grafo G_{RN} inicialmente vacío, se construyen los conjuntos C, H, E con las características, hechos y expresiones matemáticas de cada regla.

Para una regla R_i,

Paso1: Crear en el grafo el nodo correspondiente.

Paso2: A partir de los elementos c_i € C, h_i € H , e_i € E de la regla R_i.

Paso 2.1 Encontrar los caminos de navegación completos pertenecientes a c_i, h_i, e_i.

Sea C_i , H_i , E_i el conjunto de los elementos del camino de navegación completo correspondiente a las características, hechos o expresiones, si estos existen.

Paso 2.2 Determinar los tipos de elementos de estos caminos de navegación.

Si existe al menos un elemento x_i que pertenece a C_i, H_i o E_i, que no pertenece al conjunto de las tablas de la base de datos, ni a las funciones del LPT, y es x_i es un recurso de la base de datos correspondiente a la implementación de una regla R_j, crear el nodo correspondiente a R_j y la arista dirigida de R_i a R_j . R_i depende de R_j.

Si m=|C ∪ H ∪ E| y n es el número de tablas de la base de datos la complejidad temporal de este algoritmo es O (m,n).

2.6. Repositorios

Es esencial en el trabajo con reglas de negocio, el diseño del repositorio de reglas [93].

El repositorio de reglas que se propone en esta investigación se estructura en dos secciones: repositorio primario y repositorio de generación.

La creación del repositorio primario de reglas es esencial en la gestión de las reglas de negocio. En este repositorio se propone guardar para cada regla, los principales atributos inherentes a estas: *id* (identificador), *tipo de regla* para identificar la categoría de la regla [103]. Cada regla es representada de acuerdo a los tres niveles de expresión utilizados: *informal, técnico y formal*. La etiqueta *Periodos Activación* contiene el conjunto de *PeríodoActivación* que representa un período de tiempo en que la regla está activa y se describe con los atributos *id* (identificador), *fecha_inicio* y *fecha_ fin.* La abstracción del repositorio se concibe como un árbol, véase figura 2.7.

Los atributos *idpadre* y *padre_versión* representan el identificador y la versión de la regla de la cual se derivó, lo cual se explicará en el próximo epígrafe.

.

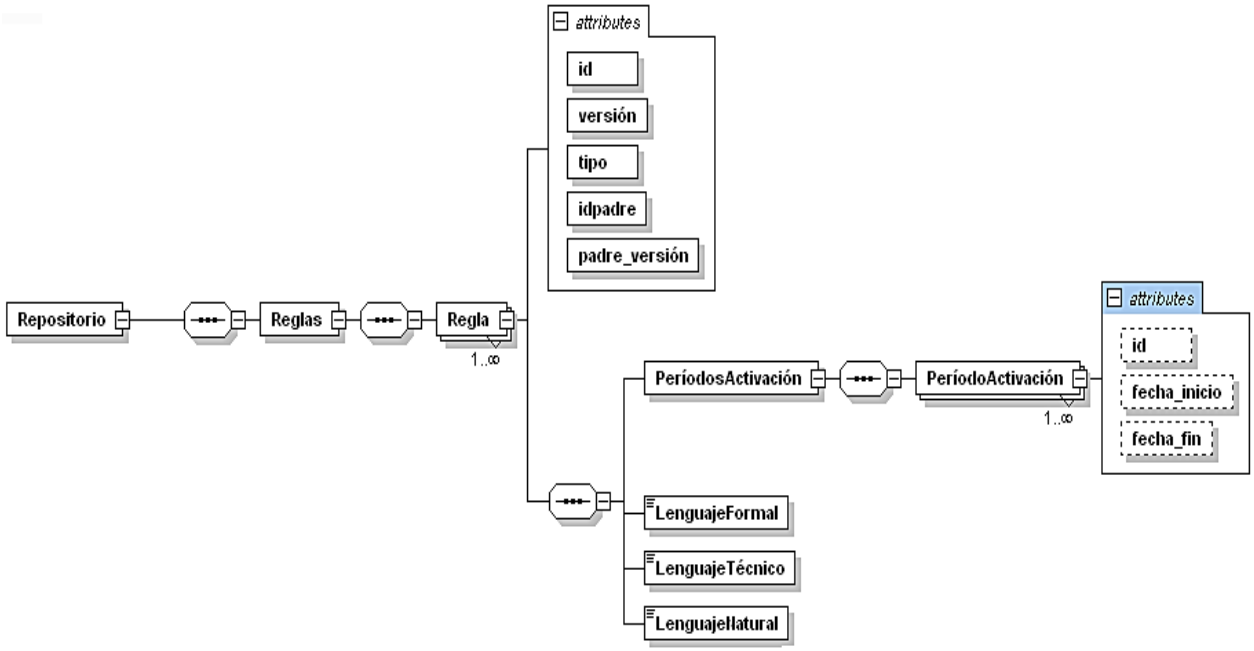

Figura 2.7 Esquema del repositorio primario de reglas

Si la regla está creada pero no está activada, no existe su implementación en la base de datos y la sección lenguaje formal está vacía. El repositorio primario de reglas se complementa con el repositorio de generación, del cual se hará referencia más adelante en este trabajo. Las reglas del repositorio de generación son un subconjunto de las reglas del repositorio primario de reglas.

2.7. Modificación de las reglas de negocios

Debido a la naturaleza dinámica de los negocios, las reglas de negocio pueden cambiar con el tiempo [42]. Ante las modificaciones de las reglas se admiten dos maneras de considerar estas modificaciones:

1. Cambiar la regla sin guardar la versión anterior.
2. Cambiar la regla guardando las versiones de la regla, en el tiempo.

Cuando una regla se modifica y pertenece a una de las categorías estudiadas en el presente trabajo, los datos que se inserten o modifiquen pueden ser tratados con la nueva regla a partir del momento del cambio. La problemática se reduce a sustituir la regla anterior por la actual y administrar los datos nuevos y los antiguos con la nueva versión de la regla ante los eventos sobre la base de datos.

En determinados casos, puede ser necesario tratar los datos con las versiones de las reglas de negocio que estaban activas cuando estos fueron creados en la base de datos. Para esto es necesario trabajar con las versiones e historial de las reglas de negocio.

2.7.1. ***Versiones de reglas de negocios***

Guardar la versión de cada regla, permite conocer la historia de las versiones: el camino de derivación. Un esquema de estas derivaciones se propone en [94], representada en forma de árbol, como se muestra en la figura 2.8. Los nodos del árbol identifican las versiones y los arcos representan las *interrelaciones de derivación* entre las versiones de reglas.

En esta investigación [119] se propone crear un árbol cuya raíz es un nodo de nivel cero denominado RN(0), así todas las reglas se derivan de este nodo. Se establece el valor cero para el atributo versión del nodo raíz; este valor nunca cambia.

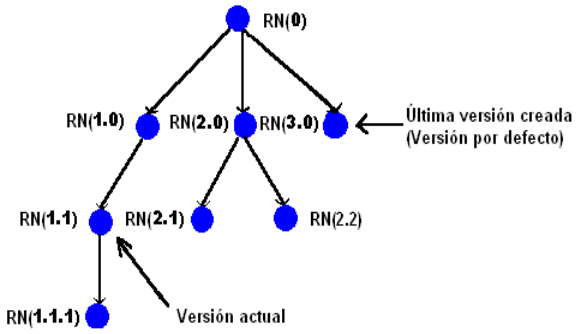

Figura 2.8 Historia de versiones de reglas

Los nodos que se encuentran en el nivel uno corresponden a las reglas que se crean antes de insertar datos en la BD, son las primeras versiones de las reglas, los que se encuentran en el nivel dos corresponden a las distintas versiones (los cambios) de las reglas del nivel uno. Se tienen tantos niveles como existan versiones de una misma regla. De esta manera se define la relación padre-hijo; por ejemplo en la figura 2.8 la regla RN (2.0) es el padre de las reglas RN (2.1) y RN (2.2) pues estas dos últimas se derivan de la regla 2.0. Se tienen tantas versiones de reglas como se necesite en el negocio.

Al insertar una regla nueva, derivada de RN(0), su atributo *padre_versión* es igual a cero; pero si esta regla se modifica, el nodo hoja tomará en *padre_versión* el valor del atributo *versión*, de la regla que esta se derivó.

2.7.2. Operaciones sobre las reglas de negocio

Las reglas, almacenadas en un repositorio, pueden ser insertadas, actualizadas y eliminadas. Las reglas deben ser implementadas por primera vez cuando pasan a estado activo; cuando se inserten datos, estos se van a regir por las reglas activas en el momento de la creación [120].

Inserción de una regla de negocio.

La inserción de nuevas reglas de negocio, significa agregar un nuevo nodo al árbol en el nivel uno como se observa en la figura 2.9.

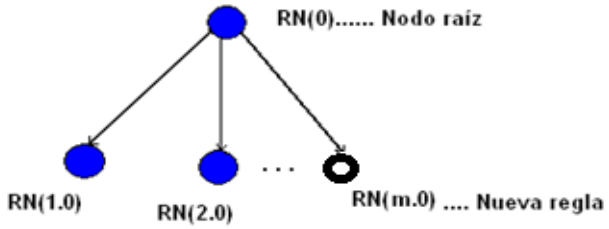

Figura 2.9 Insertar una regla nueva

Las reglas pueden hacerse activas en la BD durante uno o varios períodos de tiempo.

Actualizar una regla de negocio.

Actualizar una regla de negocio es equivalente a modificarla creando una nueva versión.

Por tanto solo se podrá modificar o actualizar las partes permitidas del patrón adoptado.

En la figura 2.10 se muestra el árbol correspondiente a un ejemplo donde se ha actualizado la regla RN(1.0), cuyo nodo se destaca con el centro transparente; esta regla al ser modificada se convierte en la "Regla anterior" y la "Regla modificada" es creada y adicionada al árbol en forma de nodo. RN(1.1), resaltado con el centro oscuro , esta regla pasa al estado de *activa* a partir del momento que se crea.

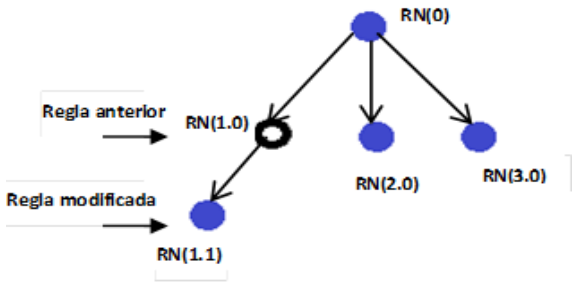

Figura 2.10 Actualización de una regla en el árbol de derivaciones

Para las reglas de tipo restricción se ha realizado un análisis exhaustivo sobre las partes del patrón de regla que pudieran modificarse para considerar que la regla se ha actualizado [121, 122] .

Eliminar una regla de negocio.

La eliminación de una regla activa puede ocasionar pérdida de información y es responsabilidad del especialista admitir esta posibilidad. En este caso se proponen dos variantes:

1. Eliminar la implementación de regla de la BD y mantener toda su información en los repositorios, se actualiza el valor del atributo *fecha_fin* correspondiente al período de activación.
2. Eliminar la regla totalmente de la base de datos y del repositorio.

   La opción más sencilla es la primera variante.

Si la regla a eliminar corresponde a un nodo que tiene derivaciones, véase figura 2.11, entonces se puede podar el árbol solo si ninguna de las versiones descendientes se encuentra activa. En caso contrario no se puede efectuar el corte. El efecto de la operación de corte es eliminar la rama a partir del nodo de la regla con un número de versión específico y eliminar todas las versiones que se derivan de ella. No se permite eliminar la versión cero o nodo raíz.

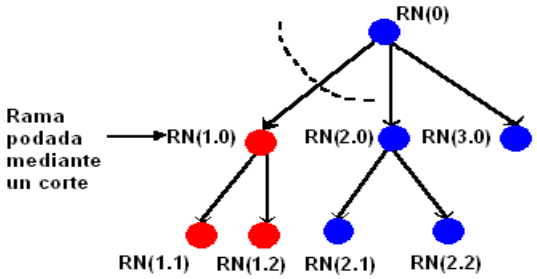

Figura 2.11 Ejecución de un "corte" en el árbol de derivación de versiones de reglas

2.7.3. Mantenimiento de las reglas de negocio de restricción y su correspondencia con los datos

Cuando una regla es modificada, usando la variante de guardar las versiones de las reglas, debe tenerse en cuenta la necesidad de conservar la coherencia de los datos con las mismas reglas utilizadas para su creación, por esto se necesita conocer cuáles tuplas de qué tablas (representantes de entidades del negocio) fueron insertadas con cuál versión de qué regla.

Se analizan diferentes variantes para realizar el mantenimiento de las reglas de negocio, con el objetivo de poder aplicar las reglas vigentes cuando se insertó cada tupla, independientemente si las versiones de estas reglas están activas o no en el momento en que los datos son evaluados. Se estudian tres maneras organizadas en dos variantes.

1. Asociación datos-tiempo.
2. Asociación datos-reglas mediante:
   a. Una bitácora.
   b. Tablas de relación tupla-regla.

Después de examinar cada una de estas variantes, sus ventajas y desventajas se selecciona la 2b, pues resulta ventajosa con respecto a la velocidad para hacer búsquedas en la base de datos [122].

2.7.4. ***Asociación datos-reglas mediante: Tablas de relación tupla-regla***

Se propone adicionar a la base de datos del negocio la tabla *Regla*, y tablas de asociación *tupla-regla*, procedentes de interrelaciones muchos-muchos (N:M) entre una regla y las tuplas insertadas bajo su acción.

Regla constituye una tabla con los identificadores de las reglas, y sus atributos principales, las interrelaciones *RelacionTabla¡Regla* conformarán tablas que tendrán la llave primaria compuesta por los atributos llaves de las tablas de la BD del negocio y de la tabla Regla.

Cuando se realiza una operación de inserción, eliminación o actualización de una regla se actualiza la tabla *Regla*, y se mantienen actualizadas las tablas de asociación tupla-regla, para cada uno de los eventos sobre las tablas implicadas.

2.8. Aspectos generales sobre la implementación de las reglas de negocio

Los requisitos de la base de datos relacional y de las reglas de negocio desde la perspectiva de los datos presentados, garantiza la implementación automática de las reglas.

Las categorías de reglas de negocio desde la perspectiva de los datos, escritas en lenguaje LPT, en función de tablas y atributos de la base de datos se implementarán como conjunto de mecanismos o recursos de bases de datos, en lenguaje SQL a manera de generación automática de estas implementaciones.

Se propone una traducción del lenguaje LPT a SQL, teniendo en cuenta que las instrucciones generadas dependerán del gestor relacional que se utilice.

La base de datos del negocio debe caracterizarse por lo siguiente:

- Las entidades del negocio relevantes se corresponden con tablas de la base de datos, o atributos de estas.
- Las tablas están interrelacionadas, con existencia de integridad referencial

La información de la base de datos se obtiene del catálogo de la base de datos sobre las tablas, atributos, llaves primarias, llaves extranjeras, funciones, vistas, disparadores. Este proceso es esencial para la adecuada generación de las reglas y su posterior ejecución.

Para cada una de las categorías de reglas en forma de patrones y expresadas en lenguaje LPT, se establecen recursos de bases de datos para implementar las reglas, que incluyen los recursos que

permiten evaluar que la regla se cumpla o hacerla cumplir y los recursos que implementan la lista de eventos.

Para implementar los patrones de reglas en forma de recursos de bases de datos diseñan vistas, funciones y procedimientos almacenados, se utilizan los que se consideran más adecuados en cada caso.

Para implementar la lista de eventos, dónde y cuándo se evalúan las reglas se utilizan los disparadores sobre las tablas del *camino de navegación completo* de las características, los hechos, o expresiones aritméticas que cumplen determinadas propiedades: en el grafo asociado al diagrama relacional constituyen nodos con aristas salientes, porque contienen llaves extranjeras a otras tablas. Los eventos de inserción, modificación, eliminación sobre las tablas correspondientes, responden a la pregunta de cuándo.

Las características específicas de las implementaciones de las reglas serán tratadas en el capítulo tres.

2.9. Conclusiones Parciales

En este capítulo se establece una clasificación de reglas de negocio desde la perspectiva de los datos, pues se evidencian categorías de reglas que pueden ser chequeadas naturalmente sobre las instancias de datos cuando se efectúan operaciones de inserción, modificación y eliminación sobre la base de datos.

Las categorías propuestas son: restricción, cálculo, notificación y clasificación; se presentan en forma de patrones de reglas, expresadas en tres niveles: informal, técnico y formal, en este último nivel se implementan como mecanismos de bases de datos.

Para la expresión de nivel técnico de los patrones de reglas se crea un lenguaje denominado LPT, inspirado en OCL y la notación punto, a partir de la necesidad de expresar los atributos de las tablas y las interrelaciones entre estas de manera sencilla. Este lenguaje se define sobre los elementos de la base de datos relacional, que refleja los términos y hechos del negocio. Se especifica la sintaxis y la semántica de este lenguaje.

Se establecen posibles interdependencias entre estas categorías de reglas, reflejo de las posibles interdependencias entre términos del propio negocio.

La traducción de las reglas escritas en LPT a mecanismos de bases de datos se concibe a través de un algoritmo que trabaja sobre un dígrafo cuyos nodos del grafo son las tablas de la base de datos y los arcos constituyen correspondencias entre llave primaria y llave foránea. Se construye

una lista de eventos para establecer el momento de chequear el cumplimiento de las reglas. Para lograr la implementación de las reglas se lee del catálogo de la base de datos, la información de tablas y atributos como una forma sintetizada del modelo de hechos.

La manera de mantener con relativa independencia la información de las reglas se logra con un repositorio de reglas el cual guarda las reglas en todas sus expresiones e información adicional necesaria.

Así, se propone una manera de actuar con las reglas de negocio que reflejan el comportamiento de sus datos, prescindiendo del uso de los SGRN para estas categorías de reglas de negocio en ámbitos de negocios donde las base de datos relacionales son componentes imprescindibles de sus SI. De esta manera se puede prescindir de la capa intermedia de objetos que usan los SGRN, que pueden tener como consecuencia bajo rendimiento y uso significativo de tiempo, por los posibles cambios frecuentes en el modelo de objetos, entre otras desventajas.

3. PROPUESTA DE IMPLEMENTACIÓN AUTOMÁTICA DE LAS CATEGORÍAS DE REGLAS DESDE LA PERSPECTIVA DE LOS DATOS

En este capítulo se establecen los recursos de bases de datos para implementar cada categoría de regla de negocio desde la perspectiva de los datos y permitir su ejecución automática. Se utilizan ejemplos de las bases de datos: trasplante renal, evaluación en secundaria básica y una BD para control de actividades de miembros de la FEU, implementadas en MS SQL Server o en PostgreSQL.

Se presenta una arquitectura para la administración de las reglas de negocio en bases de datos relacionales, las características generales del diseño e implementación de la herramienta LPT-SQL v1.3 basada en dicha arquitectura.

Se muestra el uso de la herramienta obtenida, una manera de validarla que al mismo tiempo tributa a la validación de las categorías, el lenguaje LPT y la traducción de las reglas. Se presentan dos casos de estudio en la inserción automática de reglas de negocio en bases de datos relacionales para diferentes entornos de negocios: trasplante renal e implementación de un SI para el control de comedores en la UCLV (Universidad Central ¨Marta Abreu¨ de Las Villas). Finalmente se presenta la validación del uso del conjunto de categorías de reglas de negocio desde la perspectiva de datos, que permiten la implementación automática de estas reglas en los SI operacionales.

3.1 Uso de los recursos de bases de datos.

Para lograr implementar las reglas desde la perspectiva de los datos, es necesario establecer los recursos convenientes y las maneras de hacer cumplir las reglas.

3.1.1. ***Regla de Restricción***

Para las reglas de esta categoría con el patrón:

```
<determinante> <sujeto> (no puede tener <características>) |
      (puede tener <características> solo si <hechos>)
```

se propone una vista de integridad, de manera similar a como se propone en [37, 38, 58]. Esta vista es definida sobre el <sujeto> mediante una consulta que contiene subconsultas, como

implementaciones de las <características> y los <hechos> expresados en LPT (consulta base). Los <hechos> pueden o no estar presentes en una regla tipo restricción; si no estuvieran presentes, la vista tendría la siguiente estructura general [118] .

```
CREATE VIEW VRN#? AS
 SELECT * FROM TablaX sujeto WHERE
 (/* Implementación de las <características>*/);
```

En caso que la regla incluya <hechos>, la estructura de la vista sería la siguiente:

```
CREATE VIEW VRN#? AS
 SELECT * FROM TablaX sujeto WHERE
 NOT (/* Implementación de las <características> */)
      OR ( /* Implementación de los <hechos> */);
```

Para esta variante del patrón las características se cumplen solo si se cumplen los hechos, de modo que la justificación de esta expresión lógica para la cláusula WHERE viene dado por el hecho que $P \Rightarrow Q \equiv \neg P \vee Q$.

Si como resultado de la implementación de las <características> o de los <hechos> no se obtuvieran expresiones lógicas “is not NULL” (o un equivalente que signifique diferente de nulo)) las expresiones lógicas fueran “is null”, entonces para este valor de las características no se recuperan instancias incumplidoras de la regla .

Generalmente para una operación aislada sobre la BD, cuando existen reglas de restricción implicadas, las listas de eventos se implementa con disparadores. Cada uno tiene la estructura siguiente [109]:

```
CREATE TRIGGER TIRN#? AFTER INSERT ON TablaX AS
     if (EXISTS (SELECT * FROM VRN#? ) )
     BEGIN
       raiserror('RN#?',16,1);
      rollback transaction; END
```

Para el ejemplo presentado en el epígrafe 2.1, para la regla RN#1 se tiene:

```
CREATE VIEW VRN#8 AS
  (SELECT * FROM Cirujano sujeto WHERE
  (SELECT COUNT(*) FROM Cirujano a, DonanteVivo b
      WHERE (sujeto.idCirujano = a.idCirujano) and
            (a.idCirujano = b.idCirujano)) >
                ((SELECT COUNT(*) FROM DonanteVivo) / 6));
```

Uno de los disparadores para esta regla es:

```
CREATE TRIGGER TIRN#8_1 ON DonanteVivo  FOR INSERT AS
  if (EXISTS (SELECT * FROM VRN#8))
     BEGIN raiserror('RN#8',16,1);  rollback transaction; END
```

*3.1.2. **Regla de Cómputo***

Para este patrón existen tres variantes de la regla, para cada variante (**V1**, **V2**, **V3**) abordadas en el epígrafe 2.1.2, se proponen los recursos de implementación .

V1: `<determinante><resultado> es calculado como <expresión matemática>`

Se genera una función sin parámetros que calcula el resultado.

En la generación de la regla, el cálculo no necesariamente involucra a una instancia específica de una tabla, sino a todas las instancias; para lograr esto se crea una función sin parámetros con el mismo nombre indicado en <resultado> que devuelve el valor de las operaciones definidas por la <expresión matemática>, el tipo de retorno es float, tipo de datos que puede abarcar cualquier resultado numérico. La función tiene la siguiente estructura general:

```
CREATE FUNCTION <resultado> () RETURNS float AS
  BEGIN
  DECLARE @Y float SET @Y=/ (<operación de los SELECT involucrados>)/
  RETURN @Y
```

Ejemplo: La matrícula de la escuela es calculada como sizeof(estudiante.codigo).

```
CREATE FUNCTION matricula () RETURNS float AS
  BEGIN DECLARE @y float
     SET @y = (SELECT COUNT( a.codigo ) FROM estudiante  RETURN @y END
```

```
V2: <determinante><resultado> en <sujeto> es calculado como
                  <expresión_matemática>
```

Se genera una función que calcula un resultado para una tupla, tiene como parámetros de entrada los atributos de la llave del sujeto.

La función tiene como parámetros los atributos de la llave del <sujeto>, de esta forma el cálculo correspondiente a la expresión matemática se ejecuta solo para aquella tupla que contenga estos valores de los atributos de la llave y computa el <resultado>, el tipo de retorno es float y puede ser ejecutada por el usuario o dentro del cuerpo de otra regla. Para utilizar el término del negocio que se agrega (en forma de nombre de función) para referenciar la regla en el cuerpo de otra

regla, es necesario escribir en forma de parámetro la palabra reservada sujeto. Esta función tiene la siguiente estructura general.

```
CREATE FUNCTION <resultado> (<atributos de llave del sujeto>)
  RETURNS float AS
  BEGIN
   DECLARE @Y float SET @Y=/ (<operación de los SELECT involucrados>)/
   RETURN @Y
  END
```

Ejemplo:

RN#2 El promedio_Asignatura en cursa es calculado como sujeto.promediotcp+sujeto.promedioes+max(sujeto.ASIGNATURA.EF.notaef)

```
CREATE FUNCTION promedio_Asignatura (@anombreasig text, @agradoasig
                integer ,@ecodigo integer)
RETURNS float AS
BEGIN DECLARE @y float
   SET @y = (((SELECT a.promediotcp FROM cursa a)
              + (SELECT a.promedioes FROM cursa a))
              + (SELECT MAX( c.notaef )FROM cursa a,asignatura b,ef c
         WHERE (a.anombreasig=b.nombreasig)
              AND (a.agradoasig=b.gradoasig) AND
         (b.nombreasig=c.anombreasig)
         AND (b.gradoasig=c.agradoasig)))
    RETURN @y END
```

V3: determinante><resultado> en <sujeto> para <atributo> es calculado como <expresión matemática>

En esta variante el resultado de cálculo debe ser almacenado en un atributo de una tabla de la BD, este resultado debe ser actualizado ante la ocurrencia de algún evento de inserción, actualización o eliminación. Para esta variante del patrón se generan los siguientes recursos:

- Una función con nombre dado por <resultado>, que recibe como parámetro los elementos que conforman la llave primaria de la tabla <sujeto>.
- La vista que contiene las filas de la tabla <sujeto>, donde el valor de la columna <atributo> difiere del valor definido por la función <resultado> para los valores de los atributos de la llave primaria.
- Un procedimiento almacenado, verifica que la vista que extrae los sujetos incumplidores de la regla,está vacía; en caso de no estarlo actualiza el <atributo> de

las filas de la tabla <sujeto> para las instancias donde el valor de <atributo> no coincide con el valor de <resultado>.

- Los disparadores de inserción, actualización y eliminación correspondientes para implementar la lista de eventos. Estos se encargan de ejecutar los mecanismos necesarios cuando se realiza una operación sobre la BD que pueda modificar el valor de <resultado> en el <sujeto>.

Estos recursos tienen las siguientes estructuras generales.

Función:

```
CREATE FUNCTION <resultado> (parámetros: valores de los atributos de la
                            llave del sujeto)
  RETURNS float AS
  BEGIN DECLARE @Y float SET @Y=/ (<operación de los cálculos
                                    involucrados>)/
  RETURN @Y END
```

El tipo de retorno de la función coincide con el tipo del <atributo> donde se va a insertar el <resultado>.

Vista:

```
CREATE VIEW <nombre de la vista> AS
     SELECT <cuerpo del select>
     /**Selecciona las filas cuyo atributo <resultado>difieren del
                            valor de la función**/
```

Procedimiento:

```
CREATE PROCEDURE <nombre del procedimiento> AS
 UPDATE <sujeto>  SET <atributo> = <resultado>
     WHERE EXISTS <consulta select>
     /**Realiza un acople entre las filas de la vista y la tabla
     sujeto, se actualiza donde sea verdadera la condición del acople
     **/
```

Al procedimiento almacenado se le llama PROCRN#? según la regla.

Disparadores:

```
CREATE TRIGGER <nombre>
AFTER <evento>ON <tablas>[OF <columna>]    /**nombre de cada tabla
implicada columna si es necesario para UPDATE*/
FOR  <evento> /**eventos de Update, Insert o Delete**/  AS
BEGIN EXEC <nombre del procedimiento>  END
RETURN NEW; END;  $$ LANGUAGE plpgsql;
```

Los disparadores son nombrados TIRN#? para el evento de inserción, TURN#? para la actualización y TDRN#? para la eliminación. Estos recursos tienen la siguiente estructura general.

Ejemplo para una base de datos para el control del trabajo FEU.

El LUGAR en Criollos para lugar es calculado como sujeto.edicion+100

```
CREATE OR REPLACE FUNCTION LUGAR (edicion integer) RETURNS real AS $$
          DECLARE y float;
          BEGIN
            y=(((SELECT a."edicion" FROM public."criollos" a
                 WHERE ($1=a."edicion")) + 100))  ; RETURN y;
           END;  $$LANGUAGE plpgsql;

CREATE VIEW VRN7 AS
     SELECT edicion FROM criollos
          WHERE ((lugar <> LUGAR ( edicion )) or lugar is NULL);
CREATE OR REPLACE FUNCTION PROCRN7() RETURNS trigger AS
     $$ BEGIN UPDATE criollos SET lugar = LUGAR ( edicion )
        WHERE EXISTS (SELECT edicion FROM VRN7 a
                      WHERE  a.edicion = edicion);

CREATE TRIGGER TIRN7_1 AFTER  INSERT  ON criollos
          FOR EACH ROW EXECUTE PROCEDURE PROCRN7();
CREATE TRIGGER TURN7_1 AFTER  UPDATE  ON criollos
          FOR EACH ROW EXECUTE PROCEDURE PROCRN7();
```

3.1.3. ***Regla de Notificación***

Para realizar la implementación es necesario conocer cómo se pretende alertar al usuario cuando determinados <hechos> son satisfechos. La notificación se trata mediante mensaje directo al sistema, lo que puede hacerse utilizando vistas y los disparadores para chequear el cumplimiento de la regla que en este caso no rechaza la operación si no se cumple, sino alerta una situación que es de importancia para el conocimiento del negocio.

```
Notificar <mensaje> si <hecho>
```

La estructura general de la implementación es:

```
CREATE VIEW VRN# AS
  /*consulta sobre los elementos de las tablas que cumplen los
<hechos>*/
```

```
CREATE TRIGGER <nombretrigger> AFTER   <INSERT|UPDATE|DELETE >   ON
  <tablas de los hechos>   if /* consulta sobre la vista devuelve
  elementos  se lanza la notificación*/
```

Ejemplo:

Notificar 'porciento de aprobados muy bajo' si (100 * sizeof(estudiante.promedio> 60) / sizeof(estudiante.codigo))<65

```
CREATE TRIGGER TURN#19_1 ON ESTUDIANTE FOR UPDATE AS
  if (EXISTS(SELECT * FROM VRN#19 ) )
    BEGIN raiserror ('Porciento de aprobados muy bajo',10,1); END

CREATE VIEW VRN#19 AS SELECT * FROM ESTUDIANTE WHERE
  (((100 * (SELECT COUNT( a.PROMEDIO ) FROM ESTUDIANTE a
           WHERE (a.PROMEDIO > 60 ))>0)/(SELECT COUNT(a.CODIGO)
                                        FROM ESTUDIANTE a)   )<65);
```

3.1.4. ***Regla de Clasificación***

El comportamiento e implementación de este patrón está asociado a preguntar si un <sujeto>, se corresponde o no con determinadas características para asociarlo a un nuevo término o nombre, que es la <clasificación>.

```
<sujeto> es definido como <clasificación> si <caracteristica>
```

Estructura general para la implementación es

```
CREATE FUNCTION clasificacion (idsujeto1:tipo) RETURNS BOOLEAN
  RETURN  ( /* consulta sobre las caracter que cumplen determinada
          condición sobre el sujeto*/);
```

Esta función recibe el (los) parámetro(s) interpretado(s) como el identificador de <sujeto> que se quiere clasificar y utilizado para realizar el acople en la cláusula WHERE. Los parámetros de esta función dependen de la cantidad de atributos que formen la llave primaria.

Ejemplo:

Un Cirujano es definido como Cirujano_Maestro si
MÁXIMO (sujeto.DonanteVivo.Evolución.Temperatura) < 37

```
 CREATE OR REPLACE FUNCTION Cirujano_Maestro   ( idsujeto character )
       RETURNS BOOLEAN AS
          $$ BEGIN RETURN ((SELECT MAX(Temperatura)
            FROM DonanteVivo a, Evolución b
                    WHERE (sujeto1 = a.idCirujano) and
               (a.idDonante_Potencial=b.idDonante_Potencial)) < 37)
```

```
END $$ LANGUAGE plpgsql
;
```

3.2. Modos de implementación de las reglas tipo restricción y notificación

Se analizan dos modos de implementación para estas categorías de reglas; un chequeo inmediatamente después que se realiza una operación aislada sobre los datos o un chequeo diferido, que se hará después de terminar una transacción que está compuesta por un conjunto de operaciones aisladas.

El modo de chequeo *inmediato*, hace que inmediatamente después de finalizada una operación, se chequee si se infringe cada regla de restricción involucrada usando un disparador, este tendrá la siguiente estructura general:

```
CREATE TRIGGER <nombretrigger> AFTER <INSERT|UPDATE|DELETE >
ON <tablas implicadas según LE>
if /* consulta sobre la vista, si devuelve elementos se lanza el
mensaje*/
```

Figura 3.1 Chequeo inmediato para proceso de integridad para regla de negocio tipo restricción.

Este método de chequeo inmediato, Figura 3.1, se menciona por [38, 47]. Aunque Zimbrao propone estos recursos para la implementación de las reglas de restricción no devela cómo reconoce el dónde y cuándo evaluar la regla. En [99] se describen los fundamentos del tratamiento inmediato, también se exponen en [103].

El modo de chequeo diferido permite chequear el cumplimiento de las reglas después de realizar un conjunto de operaciones concebidas como una transacción. Se propone que cada operación del sistema en la base de datos del negocio debe realizarse mediante un procedimiento, el cual se encarga de ejecutar esta transacción y determinar si puede ser aceptada o no. Véanse Anexo 2 y Anexo 3 para los modos de chequeo diferido para las reglas de restricción y notificación respectivamente.

3.3. Modificaciones del lenguaje formal para controlar versiones en reglas de restricción

Las propuestas de implementaciones de las reglas de restricción descritas no incluyen control de versiones, para permitir estas, se modifica el recurso *vista* con la adición del número de versión en la cláusula *where*. Véase Anexo 4. Para lograr la implementación de las modificaciones de estas reglas se proponen los siguientes pasos.

Secuencia de pasos para modificar una regla tipo restricción.

De todo lo anteriormente expuesto se obtiene la secuencia de pasos para modificar una regla de tipo restricción [122].

Paso 1: Desactivar la regla anterior de la base de datos. Se elimina(n) el(los) disparador(es) y la vista con los que se implementó la regla y actualizar el atributo *estado* con el valor cero, en la tabla *Regla*.

Paso 2: Borrar de los repositorios el(los) disparador(es) y la vista correspondiente a la regla, en caso que la misma esté desarrollada en el repositorio.

Paso 3: Actualizar el atributo *fecha_fin* correspondiente al período de activación de la regla anterior con el valor de la fecha actual, en el repositorio. Actualizar atributos de versión, (esta modificación es derivada de la regla anterior).

Paso 4: Compilar y generar la regla, modificada en el repositorio, con la(s) nueva(s) característica(s) y/o hecho(s).

Paso 5: Desarrollar la regla en el repositorio.

Consiste en guardar en el repositorio toda la información de la regla. Se actualiza el identificador, versión y el idpadre de la regla, se crea un nuevo Periodo de Activación, donde la fecha_inicio toma el valor de la fecha actual y fecha_fin toma valor NULL, mientras permanezca activa en la base de datos. La implementación de los recursos disparadores y vista son guardados, en el elemento Triggers de la Regla, mientras que la implementación de la vista queda guardada en el elemento Vistas.

Paso 6: Activar la regla en la base de datos e insertar en la tabla *Regla* la fila correspondiente.

Antes de activar la regla, se debe desarrollar la regla en el repositorio, e insertar los scripts correspondientes a los disparadores y vista en la base de datos. Se actualiza en la tabla Regla los atributos idregla, versión y el atributo estado toma valor 1, quedando restringido el negocio por la regla que ha sido activada.

Cada vez que se produzca un evento de inserción, eliminación o actualización de los datos, se deben actualizar las tablas de relación y realizar estos mismos eventos, para mantener los datos asociados a la versión de regla que les corresponde.

Paso 7: Si existen datos que violan la nueva versión de la regla, mostrarlos y que el experto del negocio elija qué desea hacer: eliminarlos todos, eliminar algunos y el resto conservarlos o conservarlos todos.

Paso 8: Si selecciona *eliminarlos todos o algunos:*

a) Realizar la eliminación correspondiente, el resto se *conserva* en la BD asociados a la versión de la regla que estaba activa en el momento que fueron insertados.

b) Se chequea la tabla *sujeto* y todo el camino de navegación, para evaluar solo aquellos datos que estén asociados a esa versión de la regla.

3.4. Operaciones sobre las reglas y su incidencia en los datos

Se analiza la repercusión en los datos, de las operaciones en las reglas de negocio (inserción, actualización y eliminación) [122].

Para los datos que violan las reglas se analizan tres opciones:

- eliminarlos todos,
- conservarlos,
- seleccionar los datos a eliminar y el resto conservarlos.

Para analizar cada caso se utilizan las vistas de integridad [109] que seleccionan las tuplas que no cumplen la regla.

Las consideraciones para cada una de estas opciones aparecen en [122].

Al insertar una regla en el repositorio, debe actualizarse en la BD, la tabla Reglas; las tablas *datos-reglas* correspondientes serán actualizadas al insertar datos. Véase Anexo 5.

Las restricciones asociadas al negocio pueden cambiar; y por tanto esto lleva a modificar una regla que está siendo utilizada por la BD.

Actualizar una regla de negocio tipo restricción es equivalente a modificarla, por tanto solo se podrá modificar las partes del patrón que se pueden modificar [122].

Una regla que está activa sobre la base de datos puede actualizarse. Se tienen en cuenta varios aspectos para realizar esta operación.

La descripción acerca de cómo repercuten en los datos la eliminación y la actualización de una regla aparecen en los anexos 6 y 7.

Si existen datos que entren en contradicción con la nueva versión de la regla, se procede de manera similar al caso *Insertar Regla*.

Por tanto es necesario decidir qué se va a hacer con esos datos:

1. Eliminarlos todos.
2. Conservarlos en la BD.
3. Seleccionar cuáles datos eliminar y el resto conservarlos en la BD.

Para cada una de estas variantes se analiza qué acción ejecutar con los datos que no cumplen la nueva actualización de la regla. Estas opciones deben ser decididas por funcionarios del negocio; son ellos quienes conocen plenamente, cuáles son los objetivos de la información que está contenida en la BD.

Al eliminar una regla puede ocasionar pérdida de información en el historial de reglas y se pueden perder las relaciones con los datos. Se proponen dos variantes: *Eliminar la regla de la BD conservándola en el repositorio* y *Eliminar la regla de la BD y del repositorio*, con la primera opción se conserva toda la información de la regla en el repositorio y así se permite utilizarla posteriormente.

Puede existir la posibilidad de activar una versión antigua de la regla.

Para recuperar una versión y hacerla activa, la información de esta debe conservarse en el repositorio de reglas; el atributo *fecha_fin* contiene fecha y hora en que se desactivó. Así para recuperar una versión de regla y hacerla activa, se debe:

Primero: Buscar la regla, si existe, seleccionarla, actualizarla y activarla.

Segundo: Agregar un nuevo período de activación con (id = id anterior +1) y actualizar el valor de *fecha_inicio* en el repositorio con el valor de la fecha actual.

Tercero: Insertar los recursos necesarios modificados, en la base de datos del negocio

3.5. Vista arquitectónica del proceso de traducción de las reglas de LPT a SQL

A partir de los presupuestos teóricos obtenidos se propone una arquitectura para una herramienta que permita la administración de reglas de negocio en bases de datos relacionales basada en [103] que se corresponde con la figura 3.2.

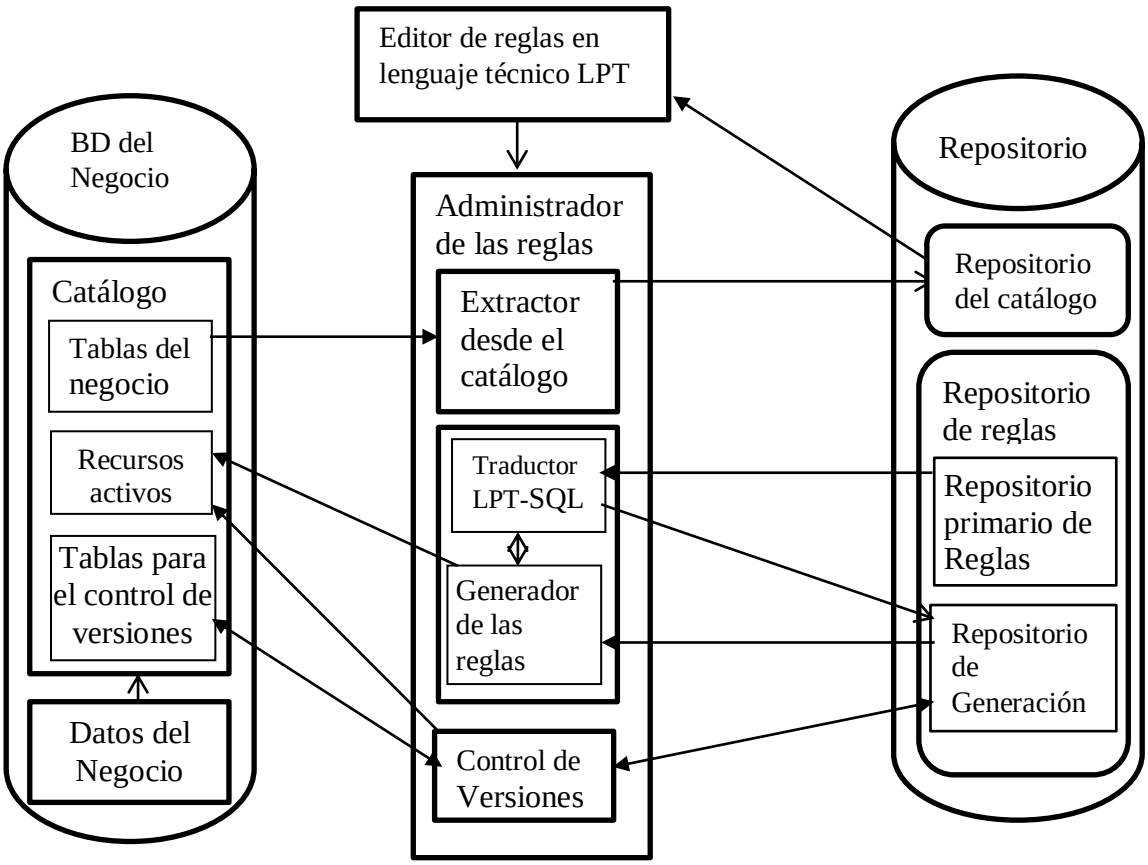

Figura 3.2 Arquitectura de la administración de reglas de negocio en BD relacionales

Los componentes principales de la arquitectura y sus subcomponentes son los siguientes:

- Repositorios.
  - Repositorio de reglas de negocio: Conformado por dos secciones, el repositorio primario de reglas que contiene cada regla en sus tres niveles: natural, técnico y formal. La segunda sección, que contiene las reglas compiladas que se deben generar y almacena la consulta base para la generación de la regla.
  - Repositorio del catálogo: Contiene información sobre las tablas de las bases de datos del negocio, información sobre los nombres, atributos, interrelaciones, recursos activos, funciones, procedimientos almacenados y funciones. También almacena información sobre las tablas auxiliares para el control de la modificación.
- Base de datos relacional del negocio: Se distinguen tres grupos de componentes:
  - Tablas principales y sus atributos: Estos han sido nombrados con los términos adecuados del negocio en cada caso.

  - Recursos activos: Los disparadores, funciones, procedimientos almacenados, muchos de los cuales constituyen implementaciones de reglas de negocios de las categorías con perspectiva de los datos.
  - Tablas de apoyo al control de versiones: Son tablas que registran con cuáles reglas se inserta cada dato.

- Editor de reglas de negocio: Es un módulo que ha de permitir editar las reglas en lenguaje natural estructurado de patrones y lenguaje LPT, reconociendo los datos de los nombres de tablas y atributos.
- Administrador de reglas: Es el núcleo de esta arquitectura, está compuesto por :
  - Módulo extractor desde el catálogo: Extrae el catálogo de la base de datos y conforma el repositorio del catálogo.
  - Traductor LPT-SQL: Con los datos almacenados en el repositorio primario de reglas, traduce las reglas de LPT a consulta base para la regla y las almacena en el repositorio de generación.
  - Generador de reglas: A partir de la información guardada en el *repositorio de generación* que constituye el núcleo de la regla en el lenguaje formal SQL produce los recursos correspondientes a la implementación de la regla y los recursos (disparadores) que implementa las listas de eventos.

3.6. La herramienta de software: LPT-SQL

Con los resultados obtenidos y a partir de la arquitectura propuesta en el epígrafe anterior se diseñó e implementó una herramienta nombrada LPT-SQL para traducir la regla en LPT de manera automática en forma de recursos activos de bases de datos relacionales. Se implementa el conjunto de pasos para la traducción de las reglas de negocio descritos en el capítulo 2 (epígrafe 2.4), en investigaciones dirigidas por la autora [108, 109] . Así se obtienen resultados prácticos a partir de los fundamentos teóricos sobre la obtención de los caminos de navegación completos, su conversión en acoples de relaciones de la base de datos para conformar la consulta base de la implementación de las regla [103].

Para la implementación de la herramienta se respeta la arquitectura propuesta. Actualmente se trabaja en el módulo de control de versiones.

El repositorio de generación se estructura como aparece en la figura 3.3

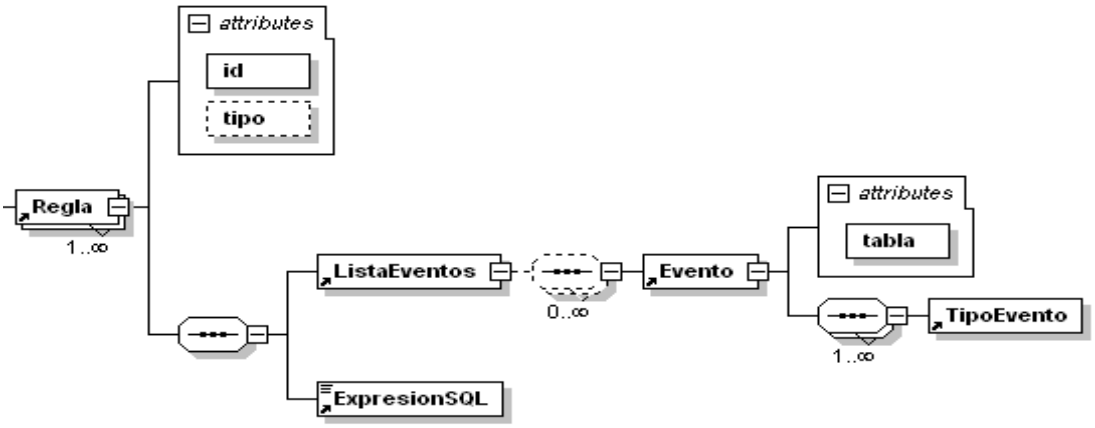

Figura 3.3 Esquema del repositorio de generación

Como se puede observar, el repositorio de generación está formado por varias reglas que son identificadas por su id y tipo.

Este repositorio ha de almacenar de cada regla una *expresión SQL*, cuerpo de implementación de la regla y la *lista de eventos* que se refiere a los eventos que pudieran infringir la regla [102], cuya representación pasa a ser de una lista binaria, a una lista de elementos en el repositorio XML donde cada elemento de *evento* tiene como atributo el nombre de una tabla y como subelemento tipo de evento, que contiene tantos elementos tipo como eventos sean necesarios chequear, pueden existir hasta tres subelementos tipo como subelementos, que se refiere al tipo de cada evento (INSERT, UPDATE, DELETE) . Esta estructura se interpreta: de ocurrir sobre esa tabla cada uno de los tipos (de evento) se puede desencadenar una violación.

Cada uno de los valores de cada atributo *tabla* es el nombre de una de las tablas del camino de navegación correspondiente, que necesitan ser evaluadas.

Este repositorio almacena información temporal y transparente para los usuarios y para la herramienta que utilice este traductor, así como para otros sistemas.

El *repositorio primario de reglas* que almacena las representaciones de las reglas queda finalmente actualizado después del proceso de generación en sus niveles de lenguaje natural, técnico y formal. Este repositorio de reglas contiene para cada regla de negocio, como aparece en la figura 2.7, los atributos inherentes a cualquier regla, id (identificador), *tipo* para identificar el tipo de regla así como la *versión*, se cuenta también con los atributos *padreversion* e *idversion*. De esta forma el repositorio acepta varios tipos de reglas.

La representación en lenguaje formal de cada regla es dividida en los recursos con que se va a implementar la regla, con la estructura que aparece en la figura 3.4. Cada recurso es identificado a partir de su tipo (vista, disparador o trigger, función etc.), y su nombre (el nombre con que va a ser implementado en la base de datos).

De esta forma se logra una representación genérica de las reglas sin importar qué recursos se utilicen para implementarlas, al mismo tiempo que se lleva un control de los recursos que van a ser implementados.

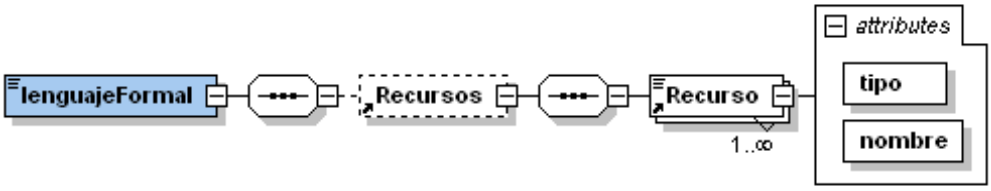

Figura 3.4 Parte del esquema XSD del repositorio de reglas referente a la representación de los recursos

Los tres repositorios usados se crean en formato XML, para lograr la portabilidad de los mismos y la interoperabilidad en casos necesarios.

Funcionalidades del sistema.

La herramienta está destinada al personal del grupo de tecnología encargado de la implementación de la base de datos o a usuarios expertos con un nivel de conocimiento básico del LPT y de la estructura del negocio. LPT-SQL, en su versión actual 1.3 debe ser utilizada inicialmente antes de insertar datos en la base de datos.

Las funcionalidades de la herramienta se pueden describir como:

- Leer el catálogo de la base de datos.
- Editar las reglas en el lenguaje LPT.
- Traducir las reglas LPT a recursos SQL.
- Generar los recursos que implementan la regla en la base de datos del negocio.

En el proceso de traducción de las reglas escritas en lenguaje técnico a recursos SQL pueden destacarse tres pasos fundamentales:

1. extracción de la regla desde el repositorio de reglas y su traducción
2. almacenamiento del código generado por el traductor en el repositorio de generación

3. generación de la regla como recursos de bases de datos y la actualización del repositorio de reglas.

Lectura del catálogo de la base de datos.

Permite recuperar todos los nombres de las tablas, atributos y recursos de la base de datos, y almacenarlos en el Repositorio del catálogo; sin crear este repositorio no se puede garantizar la escritura correcta de las reglas y su traducción.

En la figura 3.5 se muestra una parte de la estructura del repositorio del catálogo en el cual se especifica el nombre del SGBD desde el cual se debe extraer la información de las tablas, sus atributos y los mecanismos necesarios como disparadores, vistas, funciones y procedimientos almacenados.

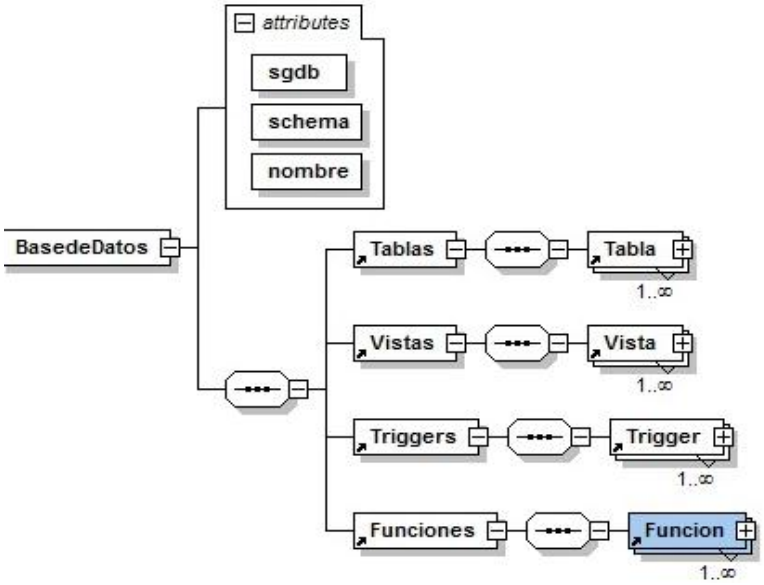

Figura 3.5 . Sección de la estructura del repositorio del catálogo

Para extraer esta información la herramienta debe establecer una conexión con la base de datos del negocio.

Traducción de la regla del LPT a SQL.

La herramienta traduce la regla escrita en LPT en una consulta o información básica que constituye la implementación de los caminos de navegación completos de la regla, en forma de consulta base.

Para que el proceso de traducción se realice adecuadamente debe tener éxito el proceso de validación sintáctica y semántica de las reglas escritas en LPT.

Es esencial que todos los requisitos sintácticos y semánticos del lenguaje sean respetados para escribir las reglas, que se resumen como indicaciones explícitas para los editores de reglas en:

- El elemento <sujeto> del patrón debe sustituirse por un término del negocio que conforma una tabla de la base de datos.
- Escribir adecuadamente los operadores de comparación, las funciones, y el resto de los elementos del lenguaje.

Los elementos de las <características> y los <hechos> deben estar compuestos por caminos de navegación válidos, lo que significa la existencia de interrelación entre las tablas, directa o indirecta, que representan términos del negocio (pudieran existir tablas intermedias entre las tablas correspondientes a dos términos del negocio).

- La evaluación de las características y los hechos deben ser valores lógicos.
- Las propiedades semánticas del lenguaje deben ser respetadas, por ejemplo los operadores de comparación deben escribirse entre elementos del mismo tipo: atributos, o valores.

En caso de violar especificaciones sintácticas y semánticas el programa emitirá una alerta de error sintáctico o semántico según el caso.

La herramienta LPT-SQL, programada en Java, contiene un compilador para las categorías de reglas de negocio desde la perspectiva de los datos, se traducen la reglas escritas en LPT a recursos SQL; la gramática se elaboró con el ANTLR, usando árboles de sintaxis abstractas, a la salida de la etapa de análisis sintáctico corresponde una estructura de tipo jerárquica del programa que se analiza [109].

3.7. Ejemplo del uso de los repositorios durante el proceso de traducción

A continuación se muestra un ejemplo de la utilización de estos repositorios tomando la siguiente regla:

RN#1:

Lenguaje Natural: Un paciente no puede ser mayor de 18 años y tener más de 5 exámenes físicos.

Lenguaje Técnico: Un paciente no puede tener Sujeto.edad>18 and sizeof (sujeto. ExamenFisico. IdExamenFisico) > 5

La información de la regla se almacena en el repositorio primario antes de compilar, en lenguaje natural estructurado y técnico; después de compilada se almacena en el repositorio de generación, y posteriormente la regla generada e implementada se guarda en el repositorio primario, así se

actualiza y completa la información sobre la misma. Estos estados de la regla se pueden observar en el Anexo 8, en las figuras A8.1, A8.2, A8.3 respectivamente.

3.8. Validación del LPT-SQL v1.2

La herramienta LPT-SQL ,versión 1.2, se validó usando la técnica de particiones o clases de equivalencia [123] y de la solución a los defectos encontrados se conforma la versión 1.3. La elaboración de las clases de equivalencia para LPT-SQL se fundamenta en la diversidad de formas en que pueden presentarse los elementos variables de los patrones de reglas escritas en LPT. La descripción de la formación de clases de equivalencia aparece en el Anexo 9. Por ejemplo, las siguientes reglas pertenecen a una misma clase, para una problemática de control del trabajo en la FEU e integralidad estudiantil:

RN#1: Un Estudiante no puede tener sujeto.edad >35

RN#6: Una Participacion_Cultutal no puede tener sujeto.numero_actos >10

Después de conformadas las clases de equivalencia, de cada una de ellas se seleccionaron las reglas más representativas dentro de su grupo y las reglas más atípicas o poco comunes donde quizás el programador olvidó chequear algún detalle. Estas reglas pasan a formar parte del grupo de entradas del software.

Resultados de la prueba.

Se detectaron 30 casos de entradas con defectos. Estos casos fueron agrupados en sus clases de equivalencia y determinar los elementos involucrados para cada clase; además se analizaron las posibilidades de interrelaciones entre algunos de los errores y defectos. Se detectaron los siguientes defectos:

1. Errores en la generación para la regla de cómputo en su tercera variante.
2. Mal funcionamiento del operador booleano *XOR.*
3. Errores en la generación de la regla de clasificación.
4. Mal funcionamiento de los operadores *Empty* y *Exists.*
5. Errores en la comparación de dos colecciones de elementos.
6. Errores en la combinación de operadores de comparación y operadores de conjunto.
7. Ausencia de chequeos semánticos.
8. Errores en el funcionamiento de la restricción mediante *acoples.*
9. Errores en la interrelación entre las reglas.

Proceso de depuración.

Este proceso de validación de la herramienta LPT-SQL sirvió para validar los fundamentos teóricos de esta investigación pues se solucionan problemas existentes en la concepción de funcionalidades del LPT, del proceso de traducción y del tratamiento de las interrelaciones entre las reglas. Así la validación por clases de equivalencia de la herramienta LPT-SQL permitió también validar el fundamento teórico del proceso desde la escritura de las reglas en LPT, su tratamiento semántico y el proceso de traducción hacia los recursos SQL.

3.9. Valoración por los expertos acerca de la investigación

Para conocer la opinión de los expertos sobre la contribución del uso de las categorías de reglas de negocio desde la perspectiva de datos y su escritura en LPT para ser implementadas automáticamente en bases de datos relacionales fue empleada como técnica de expertos, específicamente la técnica Delphi, que se basa en utilizar en la solución de problemas los juicios de un grupo de personas (expertos) con conocimientos teóricos y prácticos sobre la temática analizada, a través de un sistema de medición que permite ponderar aquellas apreciaciones cualitativas que se hayan realizado por estos expertos. El método de trabajo con expertos es utilizado actualmente en la informática y ciencia de la computación, con buena aceptación [124-126].

La opinión de los expertos sobre la contribución de la propuesta, se conoce aplicando la escala psicométrica creada por Rensis Likert en 1932 [127].

Se definieron nueve planteamientos para valorar los siguientes aspectos:

1. Utilidad de aplicar las categorías de reglas de negocio desde la perspectiva de los datos para reconocer las reglas asociadas al comportamiento de los datos.
2. Posibilidad de reconocer las categorías de reglas de negocio desde la perspectiva de los datos cuando están escritas en lenguaje coloquial del especialista.
3. Facilidad del especialista en sistemas o analista de averiguar, ante una problemática, por estos tipos de reglas.
4. Facilidad para escribir las reglas usando los patrones de las categorías desde la perspectiva de datos, en lenguaje natural.
5. Viabilidad en hacer corresponder los nombres de tablas y atributos de la base de datos del negocio con los términos del negocio en que se expresan las reglas.

6. Facilidad de escritura de las reglas usando LPT desde los patrones expresados en lenguaje natural.
7. Reconocer al lenguaje LPT como un lenguaje completo para escribir reglas de negocios desde la perspectiva de los datos.
8. Utilidad de automatizar las reglas de estas categorías desde su expresión en LPT.
9. Utilidad de reconocer y escribir reglas de negocio desde la perspectiva de los datos.

Para la evaluación se diseñó una encuesta (véase Anexo 10), la cual formula una pregunta para cada planteamiento**.** Este instrumento fue aplicado a especialistas con conocimientos y/o experiencias en bases de datos y reglas de negocios. Los expertos podían seleccionar para cada pregunta una respuesta de las siguientes, que se interpretarán con el valor escrito entre paréntesis: Muy Alto (5), Alto (4), Neutro (3), Bajo (2) y Muy Bajo(1).

Para una valoración inicial de los posibles expertos, fueron contactados profesionales de la Universidad de las Ciencias Informáticas (UCI) y las empresas cubanas Desoft, XETID, Universidad de Cienfuegos, Universidad Central "Marta Abreu" de las Villas (UCLV), Joven Club de Computación. Para asegurar la confiabilidad de las respuestas, se evaluó la idoneidad de los expertos en las temáticas sobre reglas de negocio y bases de datos a través de una encuesta (véase Anexo 11), y mediante el cálculo de su *coeficiente de competencia* [128].

Para determinar el coeficiente de competencia de los candidatos a expertos se aplicó el cálculo de dicho coeficiente de la siguiente forma:

Kcomp = ½ (Kc + Ka)

Donde:

Kcomp: Coeficiente de competencia.

Kc: Coeficiente de conocimiento o información que tiene el experto acerca del problema, calculado sobre la valoración del propio experto en una escala de 0 a 10 y multiplicado por 0,1.

Ka: Coeficiente de argumentación o fundamentación de los criterios del experto, obtenido como resultado de la suma de los puntos de acuerdo a la siguiente tabla patrón:

Las fuentes de argumentación del conocimiento de los expertos se toman de la tabla 3.1.

Tabla 3.1. Fuentes de argumentación de los expertos.

No	Fuentes de argumentación	Alto (A)	Medio (M)	Bajo (B)
1	Estudios teóricos realizados por usted (ET).	0,30	0,20	0,10
2	Experiencia adquirida durante su vida profesional (EA) .	0,50	0,37	0,30
3	Conocimiento de investigaciones y/o publicaciones nacionales e internacionales (I) .	0,05	0,04	0,03
4	Conocimiento propio sobre el estado del tema de investigación (ET) .	0,05	0,04	0,03
5	Actualización en cursos de postgrado, diplomados, maestrías, doctorado, etc (A).	0,05	0,04	0,03
6	Intuición (I).	0,05	0,03	0,02
	Total	1,00	0,70	0,50

Fuente: Tomado de [124].

Se plantea entonces que:

La Competencia de experto es Alta (A): Si Kcomp > 0,7

La Competencia de experto es Media (M): Si 0,5< Kcomp = < 0,7

La Competencia de experto es Baja (B): Si Kcomp = < 0,5

En la tabla 3.2 se puede observar el nivel de competencia de los expertos.

Tabla 3.2. Nivel de competencia de los expertos.

NroReg	ET	EA	I	CP	A	I	Ka	Kc	(ka+kc)/2	Competencia
1	0,2	0,5	0,04	10	0,05	0,05	0,84	1	0,92	Alta
2	0,3	0,37	0,04	5	0,04	0,03	0,78	0,5	0,64	Media
3	0,3	0,5	0,04	5	0,05	0,03	0,92	0,5	0,71	Alta
4	0,3	0,37	0,04	10	0,04	0,03	0,78	1	0,89	Alta
5	0,1	0,37	0,03	10	0,04	0,03	0,57	1	0,79	Alta
6	0,3	0,37	0,04	10	0,04	0,03	0,78	1	0,89	Alta
7	0,2	0,5	0,04	10	0,05	0,05	0,84	1	0,92	Alta
8	0,2	0,5	0,04	5	0,05	0,05	0,84	0,5	0,67	Media
9	0,2	0,5	0,03	5	0,05	0,03	0,81	0,5	0,66	Media
10	0,2	0,5	0,04	10	0,05	0,05	0,84	1	0,92	Alta
11	0,2	0,5	0,04	5	0,05	0,05	0,84	0,5	0,67	Media
12	0,2	0,5	0,04	10	0,05	0,05	0,84	1	0,92	Alta
13	0,2	0,5	0,04	10	0,05	0,05	0,84	1	0,92	Alta
14	0,2	0,5	0,03	5	0,05	0,05	0,83	0,5	0,67	Media

15	0,2	0,5	0,04	10	0,05	0,05	0,84	1	0,92	Alta
16	0,2	0,5	0,03	5	0,05	0,05	0,83	0,5	0,67	Media
17	0,1	0,3	0,03	10	0,03	0,02	0,48	1	0,74	Alta
18	0,1	0,3	0,03	3	0,03	0,02	0,48	0,3	0,39	Baja
19	0,2	0,37	0,03	10	0,04	0,03	0,67	1	0,84	Alta
20	0,2	0,37	0,04	5	0,04	0,03	0,68	0,5	0,59	Media
21	0,1	0,37	0,03	10	0,04	0,03	0,57	1	0,79	Alta
22	0,1	0,3	0,03	5	0,03	0,02	0,48	0,5	0,49	Baja
23	0,1	0,37	0,03	5	0,04	0,03	0,57	0,5	0,54	Media
24	0,3	0,5	0,05	10	0,05	0,05	0,95	1	0,98	Alta
25	0,3	0,5	0,04	10	0,05	0,05	0,94	1	0,97	Alta
26	0,3	0,5	0,04	10	0,05	0,05	0,94	1	0,97	Alta
27	0,2	0,5	0,04	10	0,05	0,05	0,84	1	0,92	Alta
28	0,2	0,37	0,03	10	0,04	0,03	0,67	1	0,84	Alta
29	0,2	0,37	0,03	10	0,04	0,03	0,67	1	0,84	Alta
30	0,1	0,5	0,04	10	0,05	0,05	0,74	1	0,87	Alta
31	0,1	0,3	0,05	10	0,03	0,02	0,5	1	0,75	Alta
32	0,3	0,37	0,05	10	0,04	0,03	0,79	1	0,90	Alta

Fuente: Elaboración propia.

De 32 posibles expertos identificados se desestimaron 2, cuyos índices de competencia estuvieron por debajo de 0,5. Finalmente quedaron incluidos 30 especialistas en la validación, cantidad adecuada para garantizar la confiabilidad de los resultados. Si el número de expertos seleccionados es 30 o más, el error de la decisión que se tome como resultado de la evaluación es aceptable, pues se encuentra en un rango comprendido del 1% al 2,5% [128]. De los 30 expertos, obtuvieron un coeficiente de competencia *alto* 22 y ocho de ellos, medio. A ellos se les realiza la encuesta que aparece en el anexo 11.

La composición de los expertos es reflejada en la Tabla 3.3.

Tabla 3.3. Composición de los expertos involucrados en la validación.

Perfil de trabajo	**Cantidad**	%	**Categoría Científica**	**Cantidad**	%	**Instituciones**	**Cantidad**	%
Profesores	15	50,0%	Doctores	5	16,7%	UCI	11	36,7 %
Esp Principal	2	6,7%	Máster	12	40,0%	XETID	1	3,3%
Esp A	3	10,0%	Ninguna	13	43,3%	Desoft VC	3	10,0 %
Esp B	3	10,0%	-			U Cienfuegos	6	20,0 %

Analista	5	16,7%	-			UCLV	7	23,3 %
Instructor	1	3,3%	-			Minagri	1	3,3%
Admin Redes	1	3,3%	-			JClub Comp. Placetas	1	3,3%
Total	30		**Total**	**30**		**Total**	**30**	

Fuente: Elaboración propia.

De los 30 expertos, 22 que representan un 73%, se consideran especialistas en bases de datos y sistemas de información.

La encuesta aplicada (Anexo 10) se fundamentó en los valores de la *escala de Likert* , se calcularon los porcientos de concordancia de los expertos con cada una de las posibles respuestas para los planteamientos formulados. Luego se calcula un índice porcentual (IP) que integra en un solo valor la aceptación de cada planteamiento por los evaluadores mediante la siguiente fórmula:

$$Ip = \frac{5(\%\text{ de MA})+4(\%\text{ de A})+3(\%\text{ de Neutro})+2(\%\text{ de B})+1(\%\text{ de MB})}{5}$$

En el gráfico de la figura 3.6, se puede apreciar cómo se comporta el índice porcentual para cada una de las variables que evalúan las opiniones de los expertos acerca del uso de las categorías de reglas desde la perspectiva de los datos. Para cada una de las preguntas de la encuesta se etiquetan desde u1, que corresponde a la pregunta 1,y así sucesivamente hasta u9, correspondiente a la pregunta 9.

Los índices porcentuales para todas las preguntas sobrepasa el valor de 70, siendo igual a 80 para u1, superior a 80 para u4, u6, u8 y u9. Son inferiores a 80 pero superiores a 70 para u2, u3, u5 y u7.

A partir de estos valores se puede conocer que el 100% de los expertos considera de utilidad en grado muy alto y alto, aplicar las categorías de reglas de negocio asociadas al comportamiento de los datos, con un índice porcentual de 84.

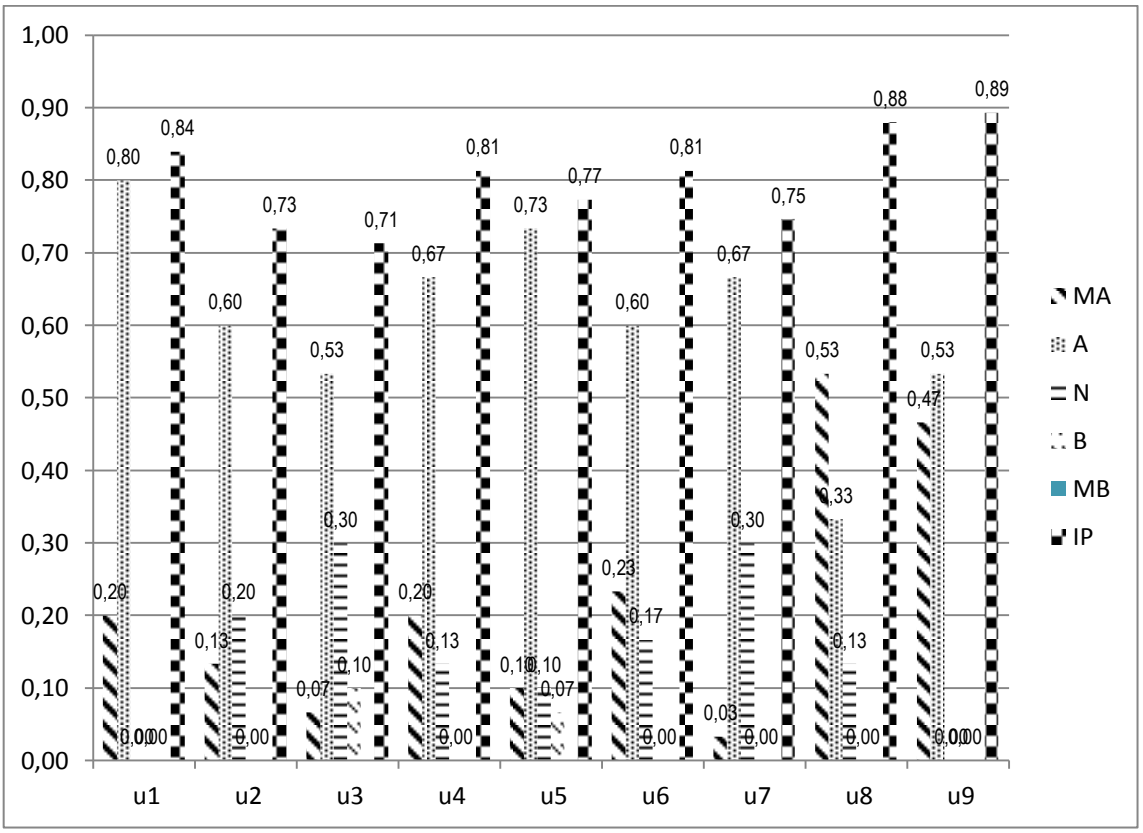

Figura 3.6 Valoración de los expertos sobre la utilidad de usar las categorías de reglas de negocio desde la perspectiva de los datos y lograr su automatización

Se evidencia también que los expertos valoran con un índice porcentual de 88 y 89, la utilidad de escribir las reglas de negocio en LPT y la utilidad de reconocer y escribir reglas de negocio con la perspectiva de los datos en el desarrollo de SI, para aquellos sistemas en los que es de esencial importancia guardar en bases de datos, los valores correspondientes a los términos del negocio.

3.10. Estudio de casos: uso de categorías de reglas de negocio desde la perspectiva de los datos

En este epígrafe se presentan dos casos: el desarrollo de una base de datos para trasplante renal y el sistema de información para control de menús en comedores universitarios.

El estudio de casos constituye un importante método para la informática en la actualidad [129-131] pues permite estudiar con profundidad una cantidad de experiencias diferentes. Para el estudio de casos, cada caso no se ve como una muestra, sino como un experimento [131]. Existen evidencias recientes de estudio de casos como método de investigación y validación en la informática [129, 132].

El primer caso seleccionado toma como punto de partida una base de datos de trasplante renal, la cual se diseñó e implementó usando las reglas de negocio asociadas a la estructura del mismo, no se utilizan de primera instancia las categorías de reglas desde la perspectiva de datos. El diseño de la base de datos y las reglas de negocio desde la perspectiva de datos no guardan relación de primera instancia.

En el segundo caso se presenta una problemática que cubre la necesidad de desarrollar un SI, particularmente un sistema de información transaccional, se recolectan las reglas de negocio haciendo énfasis en aquellas desde la perspectiva de los datos a través de la entrevista del analista con los expertos del negocio, antes de diseñar e implementar la base de datos y la interfaz del sistema.

En ambos casos el propósito es la validación de la propuesta realizada en el capítulo 2: las categorías de reglas de negocio desde la perspectiva de los datos y el uso del lenguaje LPT para utilizar los algoritmos propuestos que implementan estas reglas de manera automática dentro de la base de datos.

Estos casos fueron seleccionados en función de las necesidades del entorno en que se realiza la investigación, y en los cuales la mayoría de las entidades del negocio son modeladas en la base de datos. En estos casos se pueden encontrar diferentes tipos de reglas de negocio asociados al comportamiento de los datos guardados en bases de datos relacionales.

Siguiendo las recomendaciones de [132], inspiradas en [130, 133], los casos se presentan en función de:

- Pregunta(s) de estudio y proposiciones: Se realizan las preguntas en las que se enfoca cada caso, como orientación de la investigación en el caso. Se incluyen las proposiciones a verificar en cada caso.
- Contexto: Se presenta el contexto del caso, comparando con otros trabajos de la literatura y se justifica la relevancia del caso. Se expone la unidad de análisis de cada caso.
- Lógica de análisis: Constituye una explicación de la lógica que relaciona los datos con las proposiciones. En esta sección, la fuente de la información presentada es la observación directa o la observación participativa según la clasificación de [131].
- Discusión: Explica los criterios para interpretar los hallazgos. En cada caso se observa cómo las propuestas conducen a los resultados esperados de una manera que puede ser

repetida luego en otras situaciones, se justifican las condiciones que permiten generalizar el uso de las propuestas teniendo en cuenta cómo las condiciones de cada experimento limitan o no esta generalización.

3.11. Caso 1: Base de datos de pacientes del servicio de trasplante

El caso se inicia con el reconocimiento de las políticas del negocio para el caso de trasplante renal y el esquema lógico de la base de datos relacional correspondiente al problema, con sus tablas y atributos implementados, el catálogo de esta base de datos solamente tiene tablas y atributos, sin datos.

Este conjunto de reglas se formalizó inicialmente utilizando el sistema de categorías de Weiden, que se basa en la semántica de las mismas.

Preguntas de estudio y proposiciones.

¿Cómo las categorías de reglas de negocio desde la perspectiva de los datos pueden identificarse a pesar de que se hayan obtenido inicialmente en base a otras categorías?

¿Cómo escribir las reglas de negocio identificadas desde la perspectiva de datos en los patrones correspondientes?

¿Cómo transformar estas reglas expresadas en lenguaje natural en forma de patrones en lenguaje LPT?

¿Cómo los patrones de categorías de reglas de negocio desde la perspectiva de datos y su escritura en LPT simplifican el desarrollo de un sistema de información?

Las proposiciones correspondientes serían:

La existencia de un conjunto de categorías de reglas de negocio desde la perspectiva de los datos ayuda a identificar aquel subconjunto de reglas, del conjunto de todas las reglas enunciadas, que pueden asociarse al comportamiento de los datos.

Las reglas reconocidas como reglas desde la perspectiva de los datos, por regir el comportamiento de estos, pueden ser escritas en los patrones asociados a este conjunto de categorías.

La escritura de las reglas en lenguaje LPT debe garantizar que a cada elemento del patrón de la regla puede asociársele un término del negocio existente en la BD o un camino de navegación en que los términos en que se expresan son relaciones de una BD.

Las reglas escritas en LPT garantizan la simplificación de la codificación del sistema de información por la generación automática de la misma, en forma de recursos activos de bases de datos relacionales.

Contexto:

La unidad de análisis es el reconocimiento del comportamiento de los datos, en una base de datos de trasplante renal, a partir de todo el conjunto de reglas identificados en la etapa de análisis del sistema.

Una base de datos para sistemas de control hospitalario está encaminada a almacenar toda la información relevante de los pacientes en las entidades hospitalarias [134-136].

Un sistema para trasplante renal tiene características generales de un sistema de control hospitalario de pacientes ingresados y ambulatorios, así como características propias, en el sentido que debe almacenar información del paciente que se incluye en una lista de espera para realizársele el trasplante [137, 138]. Es útil que las bases de datos para control de pacientes tengan la capacidad de conservar la consistencia de sus reglas de negocio asociadas a los datos generales, pruebas de laboratorio, y a la validez de los valores numéricos que almacenan con posibilidades de restricciones o cálculos.

Las bases de datos para el control de pacientes se extienden en centros de atención primaria, secundaria y terciaria en numerosos hospitales [135] siendo de especial interés aquellas que almacenan datos detallados de padecimientos y tratamientos complejos y sensibles a las características específicas de los pacientes como es el trasplante renal .

Lógica del análisis:

En la lógica del análisis se abordarán las preguntas de estudio, las proposiciones y cómo estas serán respondidas.

¿Cómo las categorías de reglas de negocio desde la perspectiva de datos pueden identificarse en un conjunto de reglas a pesar de que se hayan obtenido basadas en otras categorías?

Para diversos SI, se reconocen los requisitos del sistema, muchos de estos basados en los requisitos del negocio que se originan fundamentados en las reglas de negocio [1] . Para el sistema de control de trasplante renal se identificaron un conjunto de reglas usando las categorías de Weiden [6] , estas reglas aparecen en el anexo 12, en la primera columna de la tabla y es resultado de [139]. Por otra parte en [139] aparece el esquema relacional de esta base de datos, en este caso los nombres de las tablas y los atributos en idioma inglés, equivalen en significado a los

términos (en español) de las entidades del negocio. El esquema lógico de esta base de datos, refleja aquellas reglas asociadas a la forma de relacionarse las entidades del proceso de trasplante renal que en última instancia el sistema debe almacenar. El esquema de la base de datos asume reglas asociadas a la estructura de la problemática como:

1. Un receptor puede ser asociado a un grupo de donantes potenciales.
2. Todo donante potencial puede estar asociado a un receptor.
3. Un paciente puede ser asociado a un grupo de antecedentes patológicos.
4. Todo antecedente patológico tiene asociado un paciente.
5. Un paciente puede ser asociado a un grupo de donantes potenciales.
6. Todo donante potencial tiene asociado a un paciente.
7. Un paciente puede ser asociado a un grupo de exámenes físicos.
8. Todo examen físico está asociado a un paciente.
9. Un paciente puede ser un receptor. 10.
10. Una enfermedad puede estar en un grupo de antecedentes patológicos.

Del conjunto de reglas de negocio identificadas inicialmente, pueden observarse reglas asociadas al comportamiento de los datos y sus relaciones como reconocen en alguna medida [10, 22], que se reconocen en esta investigación como *reglas de negocio desde la perspectiva de los datos* y estas reglas usan los términos del negocio que constituyen nombres de las relaciones o tablas y de los atributos correspondientes. De la problemática de trasplante renal pueden identificarse aquellas que hablan de las restricciones entre entidades del negocio como reglas de restricción.

En este caso se pueden identificar 39 reglas que aparecen en el anexo 12, a continuación se listan siete de las reglas que aparecen como políticas prohibitorias del negocio.

1. Un donante *no puede* aparecer dos veces como donante.
2. Un receptor *no puede* aparecer dos veces como receptor
3. Un receptor *no puede* ser un donante.
4. Un donador *no puede* estar en más de una operación.
5. Un acto quirúrgico de un donador *no puede* durar más de dos horas.
6. Un acto quirúrgico de un receptor *no puede* durar más de dos horas.
7. Un Trasplante Renal no *debe realizarse cuando* el VIH del paciente es positivo.

Para expresar estas reglas como políticas prohibitorias del negocio en función del esquema relacional deben identificarse primeramente los términos en que se formulan las reglas de negocio con los nombres de tablas y atributos o sinónimos de estos[140].

¿Cómo escribir las reglas de negocio identificadas desde la perspectiva de datos en los patrones correspondientes?

Las reglas que se han reconocido como asociadas a datos a partir de su semántica, primeramente deben hacerse corresponder con una de las categorías desde la perspectiva de datos, además a los términos en que fueron formuladas las reglas inicialmente se les ha asociado un nombre de entidad que aparece en las tablas de la BD Trasplante Renal.

Para cada regla que se identifica asociada al comportamiento de los datos se escribe la regla equivalente en función de las tablas y atributos de la base de datos, usando el patrón correspondiente, se identifican los términos y frases relacionadas al <sujeto>, las <características>, y los <hechos> , <expresión matemática>, <mensaje> <atributo>.

En la siguiente tabla se muestra para cada de las siete reglas listadas anteriormente su regla equivalente escrita en lenguaje de natural de patrones de reglas desde la perspectiva de los datos. Nótese que para una regla formulada originalmente, cuando se va a escribir en los *patrones de reglas desde la perspectiva de los datos*, puede suceder que el término correspondiente al <sujeto>, no aparezca con este mismo nombre en la base de datos, sino es un sinónimo, por ejemplos para la primera regla se observa como la tabla con la información de *donante* se le llama en la base de datos *historia clínica del donante*. De esta forma se realiza el proceso descrito en la tabla 3.4.

Tabla 3.4. Correspondencia entre la regla original y su escritura en el patrón correspondiente a las categorías desde la perspectiva de los datos

Regla Original	**Equivalencias en la BD**	**Regla Equivalente**
1- Un donante no puede aparecer dos veces como donante.	Un *donante* equivale a la tabla *historia clínica del donante.*	La historia clínica de un donante no puede aparecer más de una vez en las historias clínicas de los donantes

Regla Original	**Equivalencias en la BD**	**Regla Equivalente**
2- Un receptor no puede aparecer dos veces como receptor.	Un *receptor* equivale a su tabla *historia clínica del receptor*.	La historia clínica de un receptor no puede aparecer más de una vez en las historias clínicas de los receptores.
3- Un receptor no puede ser un donante.	Un *donante* y un *receptor* equivalen a las tablas de sus respectivas historias clínicas	La historia clínica de un receptor no puede aparecer como la historia clínica de un donante.
4- Un donante no puede estar en más de una operación.	Un *donante* equivale a la *historia clínica del donante*. *Operación* equivale a la tabla *trasplantes de los donadores vivos.*	La historia clínica de un donante no puede aparecer más de una vez dentro de los trasplantes de los donadores vivos.
5- Un acto quirúrgico de un donante no puede durar más de dos horas.	Un *donante* equivale a la *historia clínica del donante*. *El acto quirúrgico de un donante* equivale a la tabla *operación del donador*.	La operación de un donador no puede durar más de dos horas.
6- Un acto quirúrgico de un receptor no puede durar más de dos horas.	*El acto quirúrgico de un receptor* equivale a la tabla *operación del receptor*.	La operación de un receptor no puede durar más de dos horas.
7- Un Trasplante Renal debe realizarse cuando el VIH del paciente fue negativo.	El *paciente* equivale a la tabla *historia clínica del receptor*. Existe una tabla donde se registra los tipos de análisis realizados a los pacientes	La historia clínica de un receptor no puede tener el resultado positivo en análisis de VIH.

Fuente: [139].

En la tabla anterior se realiza la conversión de la regla del negocio en su escritura original a lenguaje natural estructurado de patrones. Cuando se cuenta con alguna regla que se identifica como desde la perspectiva de los datos y no aparecen términos equivalentes en la base de datos, se asume que estos términos no necesitan hacerse persistentes en la BD, por tanto no será evaluada ante operaciones sobre los datos del negocio, por tanto no se confirma como una regla desde la perspectiva de los datos. Véase anexo 12.

Para formular la regla en función de los elementos de los patrones correspondientes, se puede organizar como se muestra a continuación para estas reglas de restricción:

La historia clínica de un donante no puede aparecer más de una vez en las historias clínicas de los donantes.

<Sujeto>: historia clínica de un donante *<Características>:* aparecer más de una vez en las historias clínicas de los donantes.

La historia clínica de un receptor no puede aparecer más de una vez en las historias clínicas de los receptores.

<Sujeto>: historia clínica de un receptor *<Características>:* aparecer más de una vez en las historias clínicas de los receptores.

La historia clínica de un receptor no puede aparecer como la historia clínica de un donante.

<Sujeto>: historia clínica de un receptor *<Características>:* aparecer como la historia clínica de un donante.

¿Cómo escribir estas reglas expresadas en lenguaje natural en forma de patrones en lenguaje LPT?

Para escribir las reglas en LPT, es necesario validar que cada elemento del patrón pueda expresarse con notación punto, por lo tanto debe cumplirse lo siguiente:

Debe existir un nombre de tabla en la base de datos equivalente a un término del negocio, para el caso del <sujeto> de la regla y para los elementos <características> , <hechos>, <expresión matemática>, etc., los términos que constituyen sus caminos de navegación son nombres de tablas y/o atributos equivalentes a los términos principales del negocio que están contenidas en las reglas. Para los tres primeros ejemplos descritos, considerando que las tablas se nombran en idioma inglés, se tienen las reglas R1, R2 y R3 en LPT, con las tablas e interrelaciones correspondientes en las figuras 3.7, 3.8 y 3.9 respectivamente.

R1: Lenguaje Técnico: Un clinical_histories no puede tener sizeof(sujeto.clinical_history_of_donors.clinical_history_id)>1 . (Véase figura 3.7).

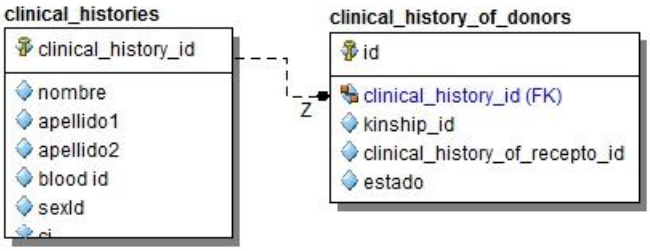

Figura 3.7. Tablas involucradas en la Regla 1

R2 Lenguaje Técnico: Un clinical_histories no puede tener sizeof(sujeto.clinical_history_of_receptors.clinical_history_id)>1. (Véase figura 3.8).

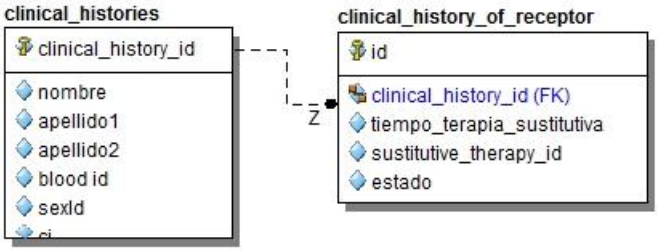

Figura 3.8. Tablas involucradas en la Regla 2

Regla 3: ***Lenguaje Técnico:*** Un clinical_histories no puede tener sujeto.clinical_history_of_receptors.clinical_history_id= sujeto.clinical_history_of_donors.clinical_history_id . (Véase figura 3.9)

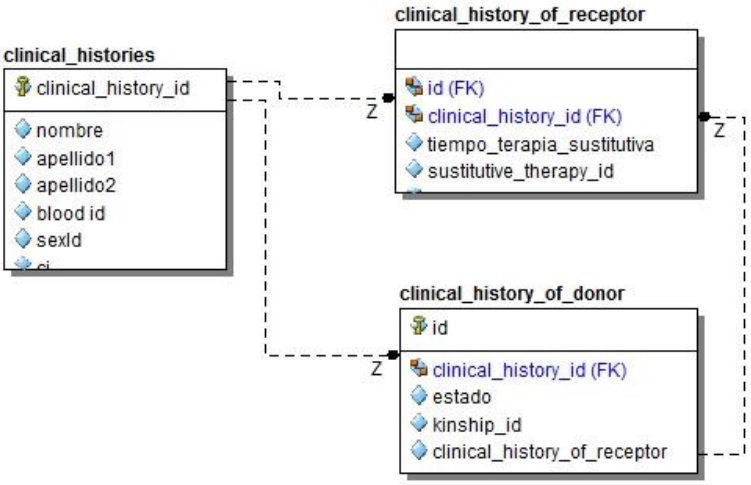

Figura 3.9 Tablas involucradas en la regla 3.

¿Cómo los patrones de categorías de reglas de negocio desde la perspectiva de datos y su escritura en LPT simplifican el desarrollo de un sistema de información?

Para cada regla escrita en LPT se genera un conjunto de recursos activos de la base de datos, lo que ahorra tiempo de codificación del programador.

Si no se utilizaran las categorías desde la perspectiva de los datos para implementar las reglas desde la perspectiva de datos en las bases de datos del negocio, y el LPT, el personal técnico necesita adicionar a sus habilidades de diseñador de datos las de programador de recursos activos de bases de datos para implementar las reglas a mano dentro del código.

Discusión:

Como se puede observar en el caso de un sistema para trasplante renal, se conoce el grupo de reglas de negocio extraídas desde la entrevista con el usuario y clasificadas por el analista previamente según un conjunto de categorías reconocidas por Weiden [52] . Por otro lado se cuenta con el diseño e implementación del esquema lógico de la base de datos, y no existen evidencias de cómo se ha diseñado la base de datos; lo que se revela es que el catálogo de la base de datos del negocio contiene solamente tablas y atributos, los cuales tienen una complejidad adicional: están en idioma inglés.

Se demuestra que a partir del universo de reglas de esta problemática se reconoce un subconjunto que está asociado al comportamiento de los datos almacenados en la base de datos, así ellas deben ser evaluadas ante las operaciones de inserción, actualización y eliminación sobre los datos, por tanto son reglas desde la perspectiva de los datos.

En este caso el subconjunto de reglas asociadas al comportamiento de los datos se reconoce en un paso posterior a identificar todas las reglas del negocio así como después que se cuenta con el esquema relacional de la base de datos. Cuando se reconocen los términos con que se expresan estas reglas en función de las tablas y atributos de la base de datos, y además la existencia de interrelaciones entre estos, estas reglas se pueden escribir usando LPT y quedan listas para ser implementadas automáticamente dentro de las bases de datos correspondientes como recursos activos.

Se puede entonces concluir que con la utilización de los patrones de categorías de reglas desde la perspectiva de datos, para el caso de trasplante renal se pueden identificar reglas asociadas al comportamiento de los datos que quedarán implementadas dentro de la base de datos con la naturalidad [22] que proviene de que estas reglas realmente deben ser evaluadas sobre los datos

correspondientes, al realizarse operaciones sobre ellos: inserción, modificación y eliminación independientemente de la interfaz del sistema.

Debido a que la presentación del caso y su discusión pudieran ser similares con respecto a otras bases de datos para diferente problemática, es posible notar que el caso puede verse como ejemplo de uso para aplicarse en otros sistemas con intenciones similares. Este estudio de caso ha sido muy útil como herramienta de investigación algunos de estos resultados han sido publicados en [112, 139, 140].

3.12. Caso 2: Sistema de información de planificación de menús en comedores universitarios

En este caso se presenta una problemática para desarrollar un sistema de base de datos que permita planificar y guardar los menús de los comedores universitarios pues actualmente el uso de tablas Excel limita la gestión de las planificaciones de los menús, así como los reportes periódicos planificados y otros reportes ad hoc necesarios en tiempo real. A partir de la descripción del problema, realizada por el analista y basada en la entrevista con el usuario, se reconoce el conjunto de reglas de negocio y el universo de discurso para la planificación de menús.

Pregunta de estudio y proposiciones:

¿Cómo escribir las reglas de negocio desde la perspectiva de los datos en forma de sus patrones en lenguaje natural a partir de conocer el vocabulario del negocio a través de reconocer el universo de discurso en entrevistas con clientes y usuarios?

¿Cómo contribuyen las categorías de *reglas de negocio desde la perspectiva de los datos* a completar el conocimiento de las reglas sobre las entidades del negocio y las estructuras de las interrelaciones entre estas?

¿Cómo escribir las reglas de negocio identificadas desde la perspectivas de los datos en LPT a partir de la creación del diseño relacional basado en aquellas reglas que develan la interrelación estructural de las entidades del negocio y las propias reglas desde la perspectiva de los datos?

¿Cómo los patrones de categorías de reglas de negocio desde la perspectiva de datos y su escritura en LPT simplifican el desarrollo de un sistema de información?

Las proposiciones correspondientes serían:

Los patrones de *las categorías de reglas de negocio desde la perspectiva de los datos* en lenguaje natural se escriben con facilidad, si se conforma un vocabulario del negocio a partir del

conocimiento del universo de discurso, porque para cada regla enunciada el analista tiene dominio de los términos utilizados.

El diseño de la base de datos a partir del conocimiento de las reglas sobre las entidades del negocio y las estructuras de las interrelaciones entre estas se completan en muchas ocasiones al reconocer las reglas desde la perspectiva de datos que revelan el comportamiento de los mismos, pues los términos del vocabulario en que se escriben los patrones de reglas, <características>, <hechos>, etc. deben estar naturalmente interrelacionados en el esquema relacional.

La escritura de las reglas en LPT , a partir de un esquema relacional en cuyo proceso de creación han jugado un papel integrador las interrelaciones entre los términos del negocio que se develan en las reglas de las categorías desde la perspectiva de los datos, se realiza con facilidad porque las tablas que constituyen los elementos (características, hechos, expresiones matemáticas) que conforman estas categorías, están interrelacionadas naturalmente en el esquema relacional y sus nombres pertenecen al vocabulario reconocido previamente.

El uso de los patrones de categorías de reglas de negocio desde la perspectiva de datos y su escritura en LPT simplifican el desarrollo de un sistema de información pues la traducción de estas reglas se realizan automáticamente en forma de recursos SQL sin necesidad de que los analistas y programadores sean expertos en programación de recursos activos de bases de datos, disminuyendo el tiempo de codificación.

Contexto

La unidad de análisis es el reconocimiento del comportamiento de los datos, en un SI para la planificación de menús en comedores universitarios. Se reconoce este problema como un sistema de base de datos, en que a partir de la descripción del problema el universo de discurso se diseña la base de datos. En la literatura se encuentran referencias relacionadas con planificación de menús [141-143]**;** fundamentalmente asociados a instituciones hospitalarias para dietas de pacientes o con perspectivas nutricionales, no administrativas como este caso.

Una base de datos para un sistema de ayuda a la planificación de menús universitarios cuenta con un conjunto moderado de entidades del negocio a modelar, lo que si se sostiene es una necesidad de almacenar la información al menos por un año completo.

No se reconoce ningún sistema informático en uso, excepto la planificación de menús a través del Excel, en este entorno; estos ficheros actualmente se reconocen insuficientes para llevar el control que se necesita: datos sobre las normas de los posibles platos a ofertar, restricciones para

elaborar el menú; además no se cuenta con el conocimiento de los productos en el almacén para elaborar los platos de manera sistematizadas por un programa informático.

Para el sistema que se desea implementar deben reconocerse un conjunto de reglas y políticas que se establecen en el negocio y no deben ser violadas por los operarios, una de las salidas del sistema es el menú de cada tipo de comida que se elaborará para cada sesión en cada comedor de la UCLV [144].

Lógica del análisis:

En la lógica del análisis se abordarán las preguntas de estudio, las proposiciones y cómo éstas serán respondidas.

¿Cómo escribir las reglas de negocio desde la perspectiva de los datos en forma de sus patrones en lenguaje natural a partir de conocer el vocabulario del negocio a través de su universo de discurso?

A partir de la descripción del negocio, las políticas del mismo y las entrevistas con los clientes del sistema, se descubren los términos del negocio y las reglas asociadas a las estructuras de la base de datos y las reglas de las categorías de *reglas desde la perspectiva de datos*.

Las entidades fundamentales del negocio que conforman los términos del vocabulario de esta problemática, entre los que se encuentran: menú, plato, norma, producto, categoría,

A partir del reconocimiento de los términos del negocio correspondientes a sus entidades, y el vocabulario del negocio en sí, se pueden formular las reglas de negocio desde la perspectiva de datos en forma de patrones en lenguaje natural. Las reglas principales son:

- Un menú no puede tener más de 12 platos.
- La cantidad total de los ingredientes a consumir es calculada como (cantidad de productos normados)* (cantidad de comensales) * (cantidad de raciones).
- Un plato a consumir no puede tener la cantidad real de productos mayor que la cantidad de productos existentes.
- Un menú no puede tener más de un plato de la categoría 'potaje'.
- Un menú no puede tener más de un plato de la categoría 'arroz'.
- Un menú no puede tener un plato de tipo arroz y otro de tipo harina.
- Un menú no puede tener más de un plato de las categorías 'Carne', 'Pescado' o 'Ave'.

¿Cómo contribuyen las categorías de reglas de negocio desde la perspectiva de los datos a completar el conocimiento de las reglas sobre las entidades del negocio y las estructuras de las interrelaciones entre estas?

Con el reconocimiento de las entidades del negocio y la conformación de un vocabulario, y las reglas asociadas a la estructura del mismo se realiza el diseño relacional de la base de datos. Este diseño pudiera ser completado al conocer las reglas identificadas en las categorías desde la perspectiva de los datos.

Ejemplos:

Al conocer la regla *Un menú no puede tener más de 12 platos.*

Se comprueba que existe una interrelación entre la entidad menú (sujeto de la regla) y los platos (término del negocio que conforma la característica).

Al reconocerse la regla:

La cantidad total de los ingredientes a consumir es calculada como (cantidad de productos normados)* (cantidad de comensales) * (cantidad de raciones)

Los datos (cantidad de productos normados), (cantidad de comensales), (cantidad de raciones) debe ser atributos de las tablas de la base de datos, o existir sinónimos de estos términos.

¿Cómo escribir las reglas de negocio identificadas desde la perspectivas de los datos en LPT a partir de la creación del diseño relacional, basado en aquellas reglas que develan la interrelación estructural de las entidades del negocio y las propias reglas desde la perspectiva de los datos?

Las reglas de negocio identificadas desde la perspectiva de los datos a partir de la creación del diseño relacional, creado en base a reglas que develan la interrelación estructural de las entidades del negocio y las propias reglas desde la perspectiva de los datos, se escriben en LPT casi intuitivamente, pues no es más que aplicar los patrones correspondientes para los elementos *características, hechos, cálculos*, etc. utilizando la notación punto, las funciones y operadores del LPT. Desde el patrón en lenguaje natural se reconocen los nombres de tablas y atributos en función de los términos de las entidades del negocio.

En estos casos se tienen, para las reglas listadas arriba:

1. Un Menu no puede tener sizeof(sujeto.Plato_consumir.id_plato)>12

2. El total en Ingredientes_consumir para cant_producto_total es calculado como sujeto.Norma.cant_producto* sujeto.Plato_consumir.Menu.cant_comensal* sujeto.Plato_consumir.racion
3. Un Plato_consumir no puede tener sujeto.Ingredientes_consumir.cant_producto_real> sujeto.Ingredientes_consumir.Norma.Producto.cant_existente
4. Un Menu no puede tener sizeof(sujeto.Plato_consumir.Platos.Categoria(categoria='potaje'))>1
5. Un Menu no puede tener sizeof(sujeto.Plato_consumir.Platos.Categoria(categoria='arroz'))>1
6. Un Menu no puede tener sujeto.Platos.Categoria.id_categoria='c1' and sujeto.Platos.Categoria.id_categoria='c3'

Un Menu no puede tener sizeof(sujeto.Plato_consumir.Platos.Categoria(categoria='Carnes'))>1or sizeof(sujeto.Plato_consumir.Platos.Categoria(categoria='Pescados'))>1 or sizeof(sujeto.Plato_consumir.Platos.Categoria(categoria='Aves'))>1 or (sizeof(sujeto.Plato_consumir.Platos.Categoria(categoria='Carnes'))+ sizeof(sujeto.Plato_consumir.Platos.Categoria(categoria='Aves'))+sizeof(sujeto.Plato_consumir.Platos.Categoria(categoria='Pescados')))>1

¿Cómo los patrones de categorías de reglas de negocio desde la perspectiva de datos y escritas en LPT simplifican el desarrollo de un sistema de información?

Para cada regla escrita en LPT se genera un conjunto de recursos activos de la base de datos, lo que ahorra tiempo de codificación del programador.

Discusión:

Como se puede observar en el caso del sistema para planificar menús de comedores universitarios el punto de partida es la descripción del proceso del negocio a partir de la entrevista con el cliente, los documentos de políticas, y catálogos de platos y normas.

Desde la entrevista con el cliente el analista reconoce el vocabulario a partir del universo de discurso, así descubre reglas asociadas a la estructura y aquellas asociadas al comportamiento de los datos, denominadas en esta investigación: reglas de negocio desde la perspectiva de datos. A partir del conocimiento de los términos de las entidades del negocio, las reglas asociadas a la estructura y organización del mismo, así como las reglas relacionadas con el comportamiento del

negocio, se realiza el diseño e implementación de la base de datos, se escriben las reglas de las categorías desde la perspectiva de los datos en los patrones correspondientes usando LPT.

Las reglas son escritas en LPT para aplicarle el algoritmo que las traduce a recursos activos de bases de datos.

Después que las reglas son implementadas dentro de los gestores de datos y gestionadas por estos, las interfaces de entrada salida a través de valores que devuelven los recursos de implementación de las reglas pueden enviar mensajes sobre el cumplimiento o no de las reglas al usuario final.

Se demuestra que asumiendo un problema de negocio como un sistema de base de datos, es viable aplicar las categorías de reglas desde la perspectiva de datos aún desde la etapa de diseño e implementación de la base de datos [144]. Este sistema tiene características específicas para la elaboración de menús en esta universidad, que pueden extenderse a otras universidades.

3.13. Conclusiones parciales

En este capítulo se presenta una manera de implementar cada categoría de regla propuesta.

Cada patrón de regla tiene su manera de escritura y su modo de traducción automática desde LPT a mecanismos de bases de datos: se usan vistas, funciones, procedimientos almacenados y disparadores. Los recursos de bases de datos que son implementados automáticamente constituyen la implementación de las reglas y garantizan que estas reglas sean evaluadas en el momento adecuado.

Las reglas de negocio están sujetas a cambios, y aunque puedan ser cambios lentos, debe tenerse en cuenta que algunos datos pudieran incumplir con la nueva versión, se consideran diferentes variantes a tener en cuenta para estos casos de modificaciones de las reglas de restricción y notificación.

Además se analiza la posibilidad de trabajar con versiones e historial de reglas, realizándose una propuesta de transformaciones en el repositorio de reglas y en la base de datos del negocio. Se hacen las consideraciones necesarias a tener en cuenta para las operaciones de inserción, modificación y eliminación sobre un repositorio de reglas, asimismo con los procederes a seguir con los datos que incumplen las reglas.

Estas bases teóricas conformadas se complementan con el diseño de una arquitectura para implementar las reglas de negocios desde la perspectiva de los datos. La obtención de la herramienta LPT-SQL permite llevar a la práctica las categorías desde la perspectiva de los datos,

el lenguaje LPT y los procedimientos para traducir las reglas en LPT a recursos de bases de datos, en los que se logra el efecto esperado ante las operaciones sobre la BD.

La validación de la herramienta da lugar al mejoramiento de la misma, al mismo tiempo que sirve de base para mejorar los propios fundamentos teóricos del trabajo.

Se usa la técnica de estudios de casos para validar la propuesta realizada a partir de casos reales que se diferencian en la manera de abordar el problema, uno a partir de las reglas formuladas en las categorías de Weiden y el esquema lógico de la base de datos, el otro a partir del levantamiento de los requisitos del negocio y del sistema. En ambos casos se puede concluir que la propuesta realizada cumple las expectativas y se evidencia que en otros SI en ambientes relacionales también se pueden realizar tales prácticas.

Además se utiliza la prueba Delphi para la evaluación por expertos de las categorías de reglas desde la perspectiva de los datos, y su escritura en lenguaje LPT para lograr su automatización, siendo la evaluación satisfactoria para su uso en la implementación de bases de datos relacionales de SI operacionales.

CONCLUSIONES Y RECOMENDACIONES

Conclusiones

Se propone un conjunto de categorías de reglas de negocio desde la perspectiva de los datos, los formulismos para su expresión y los elementos necesarios para generar automáticamente su implementación en forma de recursos SQL, dentro del esquema físico de una base de datos relacional.

- El conjunto de categorías propuesto agrupa aquellas reglas que están asociadas directamente al comportamiento de los datos del negocio. Las categorías de reglas de negocio propuestas son: restricción, cómputo, clasificación y notificación.
- Se especifica un patrón para expresar cada categoría propuesta en lenguaje informal, lo que permite identificar las reglas y escribirlas en un lenguaje natural estructurado. Se propone el lenguaje técnico LPT, para expresar las reglas en función de las relaciones y atributos de la base de datos del negocio, lo que incluye su sintaxis y semántica. Para el nivel formal se propone traducir cada una de las categorías en recursos específicos de bases de datos como son disparadores, procedimientos almacenados, funciones.
- Para el proceso de traducción LPT – recursos SQL es necesario reconocer cómo y cuándo se evalúan las reglas, para esto se elabora un conjunto de pasos con su correspondiente algoritmo para identificar en cuáles tablas y ante cuáles eventos evaluar la regla.
- El cambio necesario de las reglas de negocio impuso la necesidad de establecer cuáles elementos de los patrones y qué modificación se realiza para considerar el cambio válido. También se determinó la forma de controlar el historial de las reglas modificadas y sus implementaciones así como la relación que ellas guardan con los datos almacenados, de manera que se puedan analizar los datos con la regla actual o con la regla que estaba en vigor cuando se insertaron los mismos a la base de datos, según la decisión de los analistas.
- Se establecen los elementos de los patrones que permiten que un cambio en una regla sea válido. Se lleva un control del historial de las reglas modificadas mediante una estructura de árbol, y se definen las posibles operaciones sobre el mismo. Se establece la relación que guardan las reglas con los datos almacenados, de manera que se puedan analizar los datos con la regla actual o con la regla que estaba en vigor cuando se insertaron los mismos a la base de datos, según la decisión de los analistas.

- Se concibió una arquitectura que implemente todo este proceso: desde la escritura de la regla hasta la implementación de la misma, pasando por el control de sus versiones. A partir de esta arquitectura se creó la herramienta LPT-SQL que capta las reglas escritas en lenguaje natural estructurado y en el lenguaje LPT de patrones técnico; mediante un traductor, las reglas expresadas en LPT sean automáticamente convertidas en recursos activos de bases de datos.
- La evaluación del uso de las categorías de reglas de negocio desde la perspectiva de datos y su implementación automática se realiza utilizando la técnica Delphi. La validación de la herramienta se realizó con la técnica de clases de equivalencia y la de estudio de casos para su puesta en práctica. Ambas evaluaciones permitieron comprobar que el uso de los fundamentos teóricos propuestos pueden ser aplicados a sistemas de información con bases de datos relacionales en diferentes contextos.

Recomendaciones.

Para dar continuidad a la actual investigación se recomienda:

1. Implementar el modo diferido para la implementación automática de las reglas de negocio desde la perspectiva de los datos, a partir de las bases teóricas establecidas en esta investigación, porque permite llevar a la práctica los fundamentos teóricos establecidos para problemáticas de negocio que deben evaluar el cumplimiento de las reglas después de realizar un conjunto de operaciones con los datos.
2. Extender las soluciones para el control de versiones de las reglas tipo restricción a todas las categorías de reglas desde la perspectiva de los datos.
3. Generalizar el uso de la identificación de las categorías de reglas de negocio desde perspectiva de los datos, a partir del diseño el esquema lógico de la base de datos, nombrando las tablas y los atributos con los nombres de los términos del negocio.
4. Implantar el uso de las categorías de reglas desde la perspectiva de datos, su escritura en patrones y con LPT, a partir de las necesidades existentes en el desarrollo de SI de bases de datos contando con un necesario adiestramiento para su uso, que debe realizarse en al menos en una sesión de trabajo.
5. Proponer indicadores para cuantificar los beneficios que se alcanzan durante la adopción del uso del conjunto de categorías desde la perspectiva de los datos.

REFERENCIAS BIBLIOGRÁFICAS

1. BRG, *The Business Rule Manifesto - The Principles of Rule Independence* . R.G.R. Business Rules Group(Eds), Versión 2.0, 2 pp, Copyright, 2006-2013, Business Rules Group, 2003. http://www.businessrulesgroup.org/brmanifesto/BRManifesto., Último-acceso:[Diciembre 2011].
2. Appleton, D.S., *Business rules: the missing link.* Datamation 30 (16), 145-150pp, 1984.
3. Nelson, M.L., J. Peterson, et al., *Transitioning to a business rule management service model: Case studies from the property and casualty insurance industry.* Information & Management. 47, 11pp, 2010. www.elsevier.com/locate/im, Último-acceso:[Enero 2011].
4. Novaković , I. and V. Deletić, *Structuring of business rules in information system design and architecture* Facta universitatis-series: Electronics and Energetics. 22, (3), 305-312pp, 2009.
5. Date, C.J., *What Not How: The Business Rules Approach To Application Development*. Libro. Addison Wesley Longman Inc(eds) 2000.
6. Martínez Busto, M.E., *Administración de reglas de negocio en el ciclo de vida de los sistemas de información*. 2013, Universidad "Marta Abreu" de Las Villas, Santa Clara, Tesis de Dr.C, 201pp, Tutore(s):L.M.D. González González, 2013.
7. Speelpenning, J., P. Daux, and J. Gallus, *Data Modeling and Relational Database Design*, Editor^Editors, Oracle Corporation: 1999.
8. Fishman, N., *Modeling Business Rules with the Object Constraint Language*. Business Rules Journal, 2003, http://www.BRCommunity.com, Último-acceso:[Septiembre 2008].
9. Fishman, N., *Server-Based Rules Enforcement*. Business Rules Journal, 2003, http://www.BRCommunity.com, Último-acceso:[Julio 2012].
10. Ataullah, A. and F. Wm, *Business Policy Modeling and Enforcement in Databases*. 2011. www.cs.uwaterloo.ca/~ataulla/unpublished/BPM_in_Databases.pdf, Último-acceso:[Diciembre 2012].
11. Gottesdiener, E., *Business Rules Show Power.* Promise Application Development Trends. EBG Consulting Inc. 4 (3), 36-42pp, 1997.
12. Ashwell, R., *Define Business Rules*, Editor^Editors, CRaG Systems. p. 7, 2006.
13. Besembel, I.M. and E. Chacón, *Objetos y reglas de negocios en la integración y automatización de procesos de producción continua*. 12*pp*, 2001, http://webdelprofesor.ula.ve/ingenieria/ibc/cleiOnRn.pdf, Último-acceso:[Febrero 2008].
14. Lowenthal, B., *Rule Enabling Applications with Oracle Business Rules*. Oracle Corporation, 21*pp*, 2005, http://webdelprofesor.ula.ve/ingenieria/ibc/cleiOnRn.pdf, Último-acceso:[Marzo 2010].
15. Matei, I., *Implementing Business Rules with Software Agents*. Department of Computer Sciences, Universidad de Tampere, Tampere, Finlandia, Tesis de Tesis de Maestría, 55pp, Tutore(s):H. Kangassalo, 2006.
16. Ross, R.G., *"The Business Rule Book: Classifying, Defining and Modeling Rules"*. Libro. B.R. Solutions(eds), Database Research Group, Boston, 394pp, Segunda Ed., 1997.
17. Von Halle, B. and R. Ross, *Business rules applied: building better systems using the business rules approach*. Libro. R. Elliot(eds), John Wiley & Sons, New York, 350pp, 2002.

18. Witt, G., *Writting Effective business rules.* Morgan Kaufmann, 2012, Último-acceso:[Marzo 2013].
19. Zoet, M., J. Versendaal, et al., *Alignment of Business Process Management and Business Rules. European Conference on Information Systems (ECIS) at AIS Electronic Library (AISEL). ECIS 2011*. 2011.
20. BRS, *BRS RuleSpeak® Practitioner's Kit.* . Business Rule Solutions,LLC. , 2004. www.rulespeak.com/es/, Último-acceso:[Enero 2010].
21. Martínez Hernández, J.L., *Introduciendo semántica en un proceso de desarrollo software a través de reglas de negocio*. Escuela técnica superior de ingenieros de telecomunicación Universidad politécnica de Madrid, Madrid, Tesis de Doctoral Tutore(s):P.M.F. José Carlos González Cristóbal, 2010.
22. Morgan, T., *Defining Business Rules* in *Business Rules and Information Systems: Aligning IT with Business Goals*, Sección del Libro Addison Wesley, 0-201-74391-4, 2002.
23. Weiden, M.R., *A Critique of the Business-Rule Approach*. Department of Social Science Informatics, University of Amsterdam, Amsterdam, Tesis de Doctorado, 2000.
24. Boley, H., A. Paschke, and O. Shafiq, *RuleML 1.0: the overarching specification of web rules.* Lecture Notes in Computer Science. 6403, (4), 162-178pp, 2010.
25. Nash, E., J. Wiebensohn, et al., *Towards automated compliance checking based on a formal representation of agricultural production standards.* Computers and electronics in agriculture. 78, (1), 28-37pp, 0168-1699, 2011.
26. Paschke, A., P. Vincent, and F. Springer, *Standards for complex event processing and reaction rules*, in *Rule-Based Modeling and Computing on the Semantic Web*, Sección del Libro Springer, 128-139. 3642249078, 2011.
27. OMG, *Object Constraint Language*. Object Management Group, Inc., 238*pp*, 2010, http://www.omg.org/spec/OCL/2.2/PDF/, Último-acceso:[Abril 2011].
28. BRCommunity.com (Eds), *A Brief History of the Business Rule Approach, 3rd ed.* Business Rules Journal. 9, (11), 2008. http://www.BRCommunity.com/a2008/b448.html Último-acceso:[Diciembre 2011].
29. Anónimo. *Disadvantages of Rule-Based Systems (Part 1)*. 2010. [en. Último-acceso:[Septiembre 2012]
30. Taylor, J., *A Realistic View of Business Rules Engines.* Business Rules Journal. 10, (8), 2009. http://www.BRCommunity.com/a2009/b492.html Último-acceso:[Agosto. 2011].
31. Hartmann, S.G., *Business Rule Management System*. 2012, http://www.hartmannsoftware.com/pub/Enterprise-Rule-Applications/brms, Último-acceso:[Enero 2013].
32. Morgan, T., *Realizing Business Rules*, in *Business Rules and Information Systems: Aligning IT with Business Goals:The technology environment* Sección del Libro Addison Wesley 0-201-74391-4, 2002.
33. Gudla, R. and A. Sachab, *Generating database scripts for executing business rules relates to enterprise software in a database runtime environment*. 4-6, 20120239680, 2012,
34. Casallas, R., C. Acero, and N. López, *From high level business rules to an implementation on an event-based platform to integrate applications. International Workshop on Vocabularies, Ontologies and Rules for The Enterprise (VORTE)*. 2005. Enschede, The Netherlands.

35. Demuth, B., *THE DRESDEN OCL TOOLKIT AND ITS ROLE IN INFORMATION SYSTEMS DEVELOPMENT*. 12*pp*, 2004, http://dresden-ocl.sourceforge.net/downloads/pdfs/BirgitDemuth_TheDresdenOCLToolkit.pdf.
36. Tedjasukmana, V.N., *Translation of OCL Invariants into SQL:99 Integrity Constraints* Hamburg, Germany Tesis de 75pp, Tutore(s):D.R. MÖLLER, D.H. WEBERPALS, and M.M. GARCIA, 2006.
37. Heidenreich, F., C. Wende, and B. Demuth, *A Framework for Generating Query Language Code from OCL Invariants.* 9, (5), 2008. http://opus.kobv.de/tuberlin/volltexte/2008/1803/pdf/ECEASST_Vol_9_2008_05.pdf, Último-acceso:[Octubre 2010].
38. Zimbrão, G., R. Miranda, et al., *Enforcement of business rules in relational databases using constraints*. *XVIII Simposio Brasileiro de Bancos de Dados/SBBD* 2003,129-141. http://homepages.dcc.ufmg.br/~laender/download/sbbd03/paper74.pdf,Último-acceso:[Noviembre 2009]
39. Ross, R., *Business rule concepts*, in *Getting to the point of knowledge*, Sección del Libro Editor^Editors Third Edition, ISBN:0941049-07-8, 2009.
40. Ross, R.G., *What Is a Business Rule?* Business Rules Journal. 11 (3), 1-4pp, 2010. http://www.BRCommunity.com/a2010/b525.html, Último-acceso:[Abril 2012].
41. Hay, D. and K.A. Healy, *Defining businessrules- what are they really? Technical Report 1.3*. 2000. http://businessrulesgroup.org/first_paper/br01c0.htm, Último-
42. Bajec, M. and M. Krisper, *Managing business rules in enterprises*. Faculty of Computer and Information Science,, University of Ljubljana, Ljubljana, Tesis de 2001.
43. Bauer, E., *The business rule approach.* University of Paderborn. 2009.
44. Ross, R.G., *Principles of the Business Rule Approach*. Libro, Addison-Wesley Professional, 2003.
45. Zachman, J.A., *A Framework for Information Systems Architecture.* IBM Systems Journal. 26, (3), 276-292pp, 1987.
46. Carver, A. and T. Morgan, *A Framework for Relating Business Constraints to Information Systems.* Enterprise, Business-Process and Information Systems Modeling. 300-314pp, 2012.
47. Date, C.J., *Constraints & Predicates: A Brief Tutorial (Part 2).* Business Rules Journal. 2, 2001. http://www.BRCommunity.com, Último-acceso:[Septiembre 2010].
48. Ross, R.G., *The Business Rule Book: Classifying, Defining and Modeling Rules, Version 4.0* Libro. I. Business Rule Solutions(eds), Business Rule Solutions, 394pp, Segunda, 1997.
49. Andreescu, A. and M. Mircea, *Managing Knowledge as Business Rules.* Informatica Economică. Academy of Economic Studies, Bucharest, Romania. 13, (4), 63-74pp, 2009.
50. Von Halle, B., *Building a Business Rule System.* Data Management Review. Part 1. Faulkner & Gray, 2001.
51. Demuth, B., *The Dresden OCL Toolkit and the Business Rules Approach.* Technische Universität Dresden. 2005.
52. Weiden, M., L. Hermans, et al., *Classification and Representation of Business Rules*. *European Business Rules Conference, June*. 2004. www.omg.org/docs/ad/02-12-18.pdf,Último-acceso:[Junio 2011]
53. Halpin, T., *Verbalizing Business Rules (part 15).* Business Rules Journal. 7, (10), 2006. http://www.BRCommunity.com/a2006/b313.html, Último-acceso:[Octubre 2011].

54. Halpin, T. and J.P. Wijbenga, *FORML 2*, in *Enterprise, Business-Process and Information Systems Modeling*, Sección del Libro Springer, 247-260. 364213050X, 2010.
55. Brodersen, R.A. and R. Lankinen, *Engine for converting data from a source format to a destination format using user defined mappings*. 2010,
56. Warmer, J., Kleppe,, *The Object Constraint Language Getting Your Models Ready For MDA*. . 2003.
57. Bajwa, I.S., B. Bordbar, and M.G. Lee, *OCL constraints generation from natural language specification*. *Enterprise Distributed Object Computing Conference (EDOC), 2010 14th IEEE International*. 2010,204-213, 1424479665 IEEE.
58. Tedjasukmana, V.N., R. MÃ–Ller, et al., *Translation of OCL Invariants into SQL: 99 Integrity Constraints*. Technical University of Hamburg, Alemania, Tesis de Maestría, Tutore(s):No refiere, 2006.
59. Santos, R. and H. Invernizzi, *OCL – Estado del Arte*. . Facultad de Ciencias y Tecnología, Universidad Nova de Lisboa, 2004, http://pwp.netcabo.pt/rmss/ficheiros/textos/OCL%20-%20Paper%20Draft.pdf. .
60. Invernizzi, H. and R. Santos, *OCL - Estado del Arte*. Faculdade de Ciencias y Tecnología, Universidad Nueva de Lisboa., 2004, http://pwp.netcabo.pt/rmss/ficheiros/textos/OCL%20-%20Paper%20Draft.pdf., Último-acceso:[Octubre 2010].
61. Cover, R., *Business Rules Markup Language*. Obtained through the Internet: http://xml. coverpages. org/brml. html,[accessed 10/1/2010]. 2002.
62. Zur Muehlen, M. and M. Indulska, *Modeling languages for business processes and business rules: A representational analysis*. Information systems. 35, (4), 379-390pp, 0306-4379, 2010.
63. Bajwa, I.S., M.G. Lee, and B. Bordbar, *SBVR Business Rules Generation from Natural Language Specification*. *AAAI Spring Symposium: AI for Business Agility*. 2011.
64. Sepúlveda, S., C. Cares, and C. Cachero, *Modelado de Características para Líneas de Producto de Software: una propuesta*. *INTERNATIONAL WORKSHOP ON ADVANCED SOFTWARE ENGINEERING*. 2012. Valparaíso.
65. Baisley, D.S.W.A.t.P., *SBVR: What Are the Possibilities?* . Business Rules Journal. 9 (3), 2008.
66. Carver, A. and T. Morgan, *Characterizing Business Rules for Practical Information Systems*, in *Enterprise, Business-Process and Information Systems Modeling*, Editor^Editors, Springer-Verlag Berlin Heidelberg 2011. p. 443-452, 2011.
67. Núñez Pérez, C.A., *Edición de reglas de negocio mediante el desarrollo de herramientas sobre un modelo de hecho genérico basado en SBVR*. Ciencia de la Computación, Universidad Central "Marta Abreu" de Las Villas, Santa Clara Tesis de Ingeniería Informática, 72pp, Tutore(s):M.E.M.C. Martínez del Busto, 2012.
68. Morgan, T., *The technology environment*, in *Business Rules and Information Systems: Aligning IT with Business Goals:*, Sección del Libro Addison Wesley, , 0-201-74391-4,, 2002.
69. Bajec, M., R. Rupnik, and M. Krisper, *Using business rules technologies to bridge the gap between business and business applications*. *Proceedings of the IFIP 16th World Computer Congress 2000. Information Technology for Business Management*. 2000. Beijing, China,8, 77-85. http://bajecm.fri.uni-lj.si/downloads/Bajec%20IFIP%202000.doc,Último-acceso:[Enero 2008]

70. Bain, M.A., *How to Use Business Rules in a MySQL Database. Using MySQL Functions to Create Two-Tier Business Savvy Applications.* 2009.
71. Date, C.J., *Introducción a los Sistemas de Bases de Datos, Séptima edición*. Libro, Addison-Wesley, México, 2000.
72. Oppel, A. and R. Sheldon, *SQL: a biginner's guide*. Libro, McGraw-Hill Profesional, 2008.
73. Melton, J. and A.R. Simon, *SQL1999: underestanding relational languaje components*. Libro, Morgan Kaufmann, 2002.
74. Date, C.J., *An introduction to Database Systems.*, Sección del Libro Editor^Editors Edition Eight Pearson Education, United State, 254-291. ISBN 0-321-019784-4, 2004.
75. MSDN, *MSDN LIBRARY VisualStudio 2008*, Editor^Editors, Microsoft Corporation: 2008.
76. Chisholm, M., ed. *How to Build a Business Rules Engine*. ed. L. Homet, et al. 2006, Morgan Kaufmann: San Francisco, United States of America.
77. Veloza, A.M.R., *Modelo de un sistema de flujos de trabajo para la automatización y gestión electrónica del proceso de investigación y generación del conocimiento de la Facultad de Sistemas de Información y Documentación de la Universidad de La Salle.* Códices. 4, (2), 1794-9815, 2011.
78. Bali, M., *Drools JBoss Rules 5.0 Developer´s Guide*. Libro. P.P. Ltd(eds), Published by Packt Publishing Ltd., 32 Lincoln Road, Olton, Birmingham, 2009.
79. Cheng, H. *Visión general de integración entre ILOG JRules y WebSphere Process Server*. 2011. [en línea] http://www.ibm.com/developerworks/ssa/websphere/library/techarticles/1002_duan/1002_duan.html. Último-acceso:[21-01-2014]
80. Group, W. *Wolman Group The Information Quality Company*. 2013. [en línea] http://www.wolmangroup.com/index.php/enfoque/nuestras-herramientas. Último-acceso:[22-01-2014]
81. Ibarra, G.M. and P. Bazán, *Análisis y comparación de plataformas BRMS a través de una prueba de concepto*. *XV Workshop de Investigadores en Ciencias de la Computación*. 2013.
82. Mazza, R.G. *Drools 5.0 Candidate Release 1*. 2009. [en línea] http://salaboy.com/tag/drools-5/. Último-acceso:[November 2012]
83. MSDN2. *MSDN Library Business Rules Engine*. 2010. [en línea] http://msdn.microsoft.com/en-us/library/aa561216.aspx. Último-acceso:[22-01-2014]
84. Ortega, J.G.G. and P.M. García, *En busca de financiación: el ecosistema de inversores y emprendedores*. Libro, Netbiblo, 2011.
85. Fowler, M., *RulesEngine*. 2009, http://martinfowler.com/bliki/RulesEngine.html, Último-acceso:[Febrero 2011].
86. Friedman-Hill, E. *Jess, the Rule Engine for the Java Platform*. 2013. [en. Enero 2014]
87. Andreescu, A.I. and M. Mircea, *Perspectives on the role of business rules in dabase design.* Database System Journal. 3, (1), 59-63pp, 2012. Último-acceso:[Abril 2013].
88. SQLServer2000, M., *Libros en pantalla de Microsoft SQL Server*, Editor^Editors, 2004.
89. Barnes, M. and D. Kelly, *Play by the rules.* BYTE. 22, (6), 98-102pp, ISSN: 0360-5280, 1997.
90. JBoss Comunity, *Drools*. 2012 http://www.jboss.org/drools/, Último-acceso:[Mayo de 2013].

91. Morgan, T., *Managing Business Rules and Models*, in *Business Rules and Information Systems: Aligning IT with Business Goals*, Sección del Libro Addison Wesley 0-201-74391-4, 2002.
92. Spreeuwenberg, S., *Rule History and Versioning (Part 1)*. Business Rules Journal. 8 (11), 2007. http://www.BRCommunity.com/a2007/b375.html,
93. Bajec, M., M. Krisper, and R. Rupnik, *A methodology and tool support for managing business rules in organisations.* Information Systems. 30, (6), 423-443pp, 2005.
94. García Pérez, A.M., *Un modelo de versiones para la construcción de software de ayuda al diseño*. Departamento de Ciencias de la Computación, Universidad Central "Marta Abreu" de Las Villas, Santa Clara, Tesis de Doctorado., 116pp, Tutore(s):D.L.G. González, 1997.
95. Spreeuwenberg, S., *Rule History and Versioning (Part 2)*. Business Rules Journal. 8 (12), 2007. http://www.BRCommunity.com/a2007/b382.html Último-acceso:[Diciembre del 2009].
96. Spreeuwenberg, S., *Rule History and Versioning (Part 3)*. Business Rules Journal. 9, (1), 2008. http://www.BRCommunity.com/a2008/b387.html
97. Pérez Alonso, A., M.B. Boggiano Castillo, and L. Díaz de la Paz, *Patrones para implementar reglas de negocio en bases de datos relacionales*. *VII Conferencia Internacional de Ciencias Empresariales CICE 2010*. 2010. Hotel Meliá Las Dunas. V.C., 978-959-250-606-0.
98. Boggiano Castillo, M.B., A. Pérez Alonso, et al., *Enfoque de reglas de negocio y sus implementaciones en bases de datos relacionales. Universidad Central "Marta Abreu" de Las Villas. Cuba* CITMA-VC, *Reconocimiento a nivel provincial* 1060/2012, 2012.
99. Pérez Alonso, A., *Reglas de Negocio en Bases de Datos Relacionales*. Departamento de Ciencia de la Computación, Universidad Central "Marta Abreu" de Las Villas, Santa Clara, Tesis de 94pp, Tutore(s):M.B.B. Castillo. and R.P. Vázquez., 2010.
100. Boggiano Castillo, M.B., M.E. Martínez del Busto, et al., *Modelo de hechos para un sistema basado en reglas de negocio en el sistema de salud.* . UCIencia 2007. Cuba. 978-959-286-005-6., 2007.
101. Boggiano Castillo, M.B., M.E. Martínez del Busto, et al., *Generating restriction rules automatically with an information system*. Revista Cubana Informática Médica, Habana, 2009, http://www.rcim.sld.cu/revista_18/articulo_18.htm.
102. Pereira Toledo, A., *Solución al problema de la cardinalidad en la generación automática de reglas de negocio en bases de datos relacionales.* Departamento de Bases de Datos, Universidad Central "Marta Abreu" de Las Villas, Santa Clara, Tesis de 92pp, Tutore(s):M.M.B.B. Castillo and L.A.P. Alonso, 2009.
103. Boggiano Castillo, M.B., A. Pereira Toledo, et al., *Traductor de reglas de negocio de lenguaje técnico LPT a lenguaje formal SQL* Revista Ciencias Matemáticas. 26, (2), 125-130pp, ISSN 0256-5374, 2012. http://intranet.dict.uh.cu/rev_mat.asp
104. Boggiano Castillo, M.B., A. Pereira Toledo, et al., *Inserción automática de reglas de negocio en bases de datos.* Revista Técnica de la Facultad de Ingeniería. Universidad del Zulia. Referenciada en Scopus. 36, (3), 9pp, 0254-0770, 2013. http://revistas.luz.edu.ve/index.php/rtz/article/view/16458,
105. Boggiano Castillo, M.B., A. Pérez Alonso, et al., *Compilador de Patrones en Reglas de Negocio.* . *Memorias XI Congreso Nacional de Matemática y Computación*. 2009. La Habana, 1728-6042.

106. Boggiano-Castillo, M.B., A. Calderón Solís , et al., *Herramienta para la generación automática de reglas de negocio desde una perspectiva de datos en bases de datos relacionales. LPT-SQL versión 1.3*, U.C. Universidad Central "Marta Abreu" de las Villas, Editor^Editors, CENDA: 2013.
107. Pérez Pedraza, R., *Extensión del traductor LPT-SQL*. CIencia de la Computación Universidad Central "Marta Abreu" de Las Villas, Santa Clara, Tesis de 96pp, Tutore(s):M.B. MSc. Boggiano-Castillo, 2012.
108. Ríos Méndez, J.F., *LPT-SQL v1.3: Herramienta para la generación automática de reglas de negocio en bases de datos relacionales*. Ciencia de la Computación, Universidad Central "Marta Abreu" de Las Villas., Santa Clara, Tesis de Licenciatura 110pp, Tutore(s):M.B. MSc. Boggiano Castillo, 2013.
109. Calderón Solís, A., *Traductor LPT-SQL para reglas de negocio en bases de datos relacionales*. Departamento de Ciencias de la Computación, Universidad Central "Marta Abreu" de las Villas, Santa Clara, Tesis de 106pp, Tutore(s):M.B. MSc. Boggiano Castillo, 2011.
110. Boggiano Castillo, M.B., A. Pérez Alonso, et al., *Notación Punto en Reglas de Negocio*. *XI Congreso Nacional de Matemática y Computación* 2009. La Habana, 1728-6042.
111. Martínez del Busto, M.E., I. Moreno Montes de Oca, and A. Rodríguez Morffi, *Business vocabulary of kidney transplant with ontological approach for a generic fact model.* Revista Facultad de Ingenieríaa Universidad de Antioquia. (53), 155-162.pp, ISSN:0120-6230, 2010. //www.scielo.org.co/scielo.php?script=sci_arttext&pid=S0120-62302010000300014, Último-acceso:[Abril 2011].
112. Boggiano Castillo, M.B. and e. al, *Validación de la funcionalidad del traductor lpt-sql para la base de datos trasplante renal IX Congreso Internacional de Informática en Salud. XV Convención y Feria Internacional Informática 2013.* 2013. La Habana, 978-959-7213-02-4. . http://www.informaticahabana.cu/taxonomy/term/1?page=8,
113. Horstmann, C.S., *Big Java: Compatible with Java 5, 6 and 7*. Libro, John Wiley and Sons, 4, 2009.
114. Paton, N.W. and O. Díaz, *Active Database Systems.* ACM Computing Surveys. 31 (1), 63-103pp, 1999.
115. Boggiano Castillo, M.B., Msc., A. Pérez Alonso, Lic , et al., *Traducción de un Patrón de Reglas de Negocio en Bases de Datos Relacionales*. *Seventh LACCEI Latin American and Caribbean Conference for Engineering and Technology (LACCEI'2009). "Energy and Technology for the Americas: Education, Innovation, Technology and Practice".* . 2009, 0-9822896-2-6. http://www.laccei.org/LACCEI2009-Venezuela/Papers/IT095_BoggianoCastillo.pdf,
116. Bumble-Bee, *Parser Generator* Editor^Editors, 2000.
117. Boggiano Castillo, M.B., A. Pereira Toledo, et al., *Inserción automática de reglas de negocio en bases de datos.* Revista Técnica de la Facultad de Ingeniería. Universidad del Zulia. 36, (3), 9pp, 0254-0770, 2013. http://revistas.luz.edu.ve/index.php/rtz/article/view/16458,
118. Boggiano-Castillo, M.B., A.P.T.A. Pérez-Alonso, et al., *Automatic insertion of business rules in databases.* Rev. Téc. Ing. Univ. Zulia.
. 36, (3), 253 - 261pp, 0254-0770, 2013. revistas.luz.edu.ve/index.php/rtz/article/view/16458/15876,

119. Díaz de la Paz, L., M.B. Boggiano Castillo, and y. otros, *Consideraciones acerca de las modificaciones de reglas de negocio en bases de datos relacionales Memorias de Compumat 2011. XII Congreso Nacional de Matemática y Computación*. 2011. Santa Clara, Cuba, 978-959-250-658-9.
120. Díaz de la Paz, L. and M.B. Boggiano Castillo, *Propuestas ante cambios generados por modificaciones en reglas de negocio. Memorias de CICE 2010. VII Conferencia Internacional de Ciencias Empresariales CICE* 2010. Hotel Meliá Las Dunas. V.C., 978-959-250-606-0.
121. Díaz de la Paz, L., M.B. Boggiano Castillo, et al., *Reglas de negocio generadas automáticamente sobre bases de datos y sus modificaciones. XIV Convención y Feria Internacional Informática* 2011.
122. Díaz de la Paz, L., *Modificación de reglas de negocio creadas automáticamente en bases de datos relacionales*. Ciencia de la Computación, Universidad Central "Marta Abreu" de Las Villas, 2011, Tesis de Maestría, 84pp, Tutore(s):M.B. Boggiano Castillo and A. Pérez Alonso, 2011.
123. Sommerville, I., *Ingeniería de software*. Libro Sexta Edición, 2002.
124. Arias Orizondo, A.C., *Modelo de madurez de tres perspectivas para evaluar y planificar la adopción de arquitecturas orientadas a servicios en las organizaciones*. CDAE Universidad de Ciencias informáticas, La Habana, Tesis de Doctorado, 182pp, Tutore(s):V.D.C. Estrada Sentí and A.D.C. Febles Estrada, 2013.
125. Febles, A., *Un modelo de referencia para la gestión de configuración en la pequeña y mediana empresa de software*. Ciencia de la Computación, Instituto Superior José Antonio Echeverría, La Habana, Tesis de Doctorado, 137pp, 2004.
126. Kellenbenz, M.J., M.S. Heil, et al., *Persistent Stored Modules (SQL/PSM)*, Editor^Editors, 1999.
127. Demuth, B., H. Hussmann, and S. Loecher, *OCL as a Specification Language for Business Rules in Database Applications. UML'01 4th Intl. Conf Unified Modeling Language*. 2001. Toronto, Ontario, Canada.
128. Rodríguez, G.D.L.C.L., *Modelo de Gestión del Conocimiento para las Áreas Económicas del Sistema de Instituciones del Ministerio de Educación Superior*. Centro de Estudios para el Perfeccionamiento de la Educación Superior (CEPES), Universidad de La Habana, 2011, Tesis de Doctorado, Tutore(s):V. Estrada Sentí and J. Febles Rodríguez, 2011.
129. Chen, H.-M., R. Kazman, and O. Perry, *From Software Architecture Analysis to Service Engineering: An Empirical Study of Methodology Development for Enterprise SOA implementation.* IEEE TRANSACTIONS ON SERVICES COMPUTING. 3, (2), 15pp, 2010.
130. Dubé, L. and G. Paré, *Rigor in information systems positivist case research: current practices, trends, and recommendations,*. MIS Quarterly. 27 (4), 597-595pp, 2003.
131. Yin, R.K., *Study Research: Desing and Methods.* Libro, SAGE Publications. International Educational and Professional Publishers, Edition 3rd., 2003.
132. Moreno Espino, M., *Patrones para incorporar proactividad en sistemas informáticos*. Facultad de Ingeniería Informática, Instituto Superior Politécnico José Antonio Echeverría, La Habana, Tesis de 172pp, Tutore(s):D.I.A.R.S. Prof. Titular, D.I.M.D.D.D. Prof. Titular, and D.J.P.M. Catedrático, 2013.
133. López, C.R., *Metodología para la sistematización de los servicios de Consultoría TI: Aplicación al sector de la Manufactura*. Departamente de Tecnología Inforática y

Telecomunicaciones, Universidad de Alicente, Alicante, Tesis de 171pp, Tutore(s):D.F.M. Pérez and D.M.D. Fernández, 2011.

134. Rubinstein, A., A. Pichon-Riviere, and F. Augustovski, *Development and Implementation of health technology assessment in Argentina: two steps forward and one step back.* International Journal of Technology Assessment in Health Care. 25, (S1), 260-269pp, 1471-6348, 2009.
135. Swaminathan, S. and G.J. Alangaden, *Treatment of resistant enterococcal urinary tract infections.* Current infectious disease reports. 12, (6), 455-464pp, 1523-3847, 2010.
136. Mahoney, J., *Type 1 Diabetes in Older Adulthood: Relationships with Technological Treatments.* 2013.
137. Chen, Y.-S. and C.-H. Cheng, *Application of rough set classifiers for determining hemodialysis adequacy in ESRD patients.* Knowledge and information systems. 34, (2), 453-482pp, 0219-1377, 2013.
138. Martínez Busto, M.E., M.B. Boggiano Castillo, and e. al., *Aplicación médica para trasplante renal usando reglas de negocio.* Revista Habanera de Ciencias Médicas 11, (1), 9pp, 1729-519X 2013. http://scielo.sld.cu/pdf/rhcm/v11n1/rhcm21112.pdf
139. Alfaro, R.A.C., *Validación de la funcionalidad del traductor lpt-sql para la base de datos trasplante renal*. Ciencias de la Computación, Universidad Central "Marta Abreu" de Las Villas, Santa Clara, Tesis de Lic., Tutore(s):M.B. Boggiano-Castillo, 2012.
140. Boggiano Castillo, M.B., A. Pérez-Alonso, et al., *Aplicación de reglas de negocio para una base de datos del servicio de Nefrología de un hospital provincial*. *XIV Convención y Feria Internacional VIII Congreso Internacional de Informática en la Salud*. 2011. La Habana, 978-959-7213-01-7.
141. Domínguez, A.G. and G.C. Morales, *RCAN.* Rev Cub Aliment Nutr. . 19, (2), 2009.
142. Salinas, J., *Nuevas modalidades de formación: entre los entornos virtuales institucionales y los personales de aprendizaje.* Estrategias de innovación en la formación para el trabajo. Madrid: Torrepunta Ediciones. 2009.
143. Universidad de los Andes. *Manual del Comedor*. [en línea] http://www2.ula.ve/dsiaportal/dmdocuments/Manual%20Comedor.pdf. Último-acceso:[Enero 2013 2014]
144. Sevilla Aguilar, D., *Sistema de información para planificar el menú en comedores UCLV utilizando LPTSQL para generar reglas de negocio*. Ciencia de la Computación. UCLV, Universidad "MArta Abreu" de Las Villas, Santa Clara, Tesis de 70pp, Tutore(s):M.B. Boggiano Castillo, 2013.
145. Choi, E.-H., T. Tsuchiya, and T. Kikonu, *Model Checking Active Database Rules* 20pp, Research Center for Verification and Semantics (CVS). National Institute of Advanced Industrial Science and Technology, Nakouji, Amagasaki, Hyogo, 2006 http://ocvs.cfv.jp/tr-data/PS06-001.pdf, Último- acceso:[Noviembre 2012].
146. Gennick, J., *SQL pocket guide*. Libro, O'Reilly Media, 184pp, 2, 2006.

ANEXOS

ANEXO 1. Sintaxis y semántica del LPT

Sintaxis

Notación

La siguiente descripción sintáctica utiliza los convenios de la XBNF, incluyendo:

1. (_):=Cualquier elemento del lenguaje.
2. **O** (_):=*un (_) opcional.*
3. **#** (_):=*cualquier número de (_) incluyendo nulo*.
4. **N** (_):=*uno o más (_)*.
5. **L**(x, (_)):= *cualquier número de (_) separados por x.*
6. **List** (_):= **L**("," , (_)).

Gramática.

regla::=restricción | cómputo | clasificación | notificación

restricción ::=determinante sujeto ‘no puede tener’ características |determinante sujeto ‘puede tener’ características ‘solo si’ hechos

cómputo ::=determinante resultado ‘es calculado como’ expresión_matemática| determinante resultado ‘en’ sujeto ‘es calculado como’ expresión_matemática | determinante resultado ‘en’ sujeto ‘para’ atributo ‘es calculado como’ expresión_matemática

clasificación ::= determinante sujeto ‘es definido como’ clasif ‘si’ características | determinante sujeto ‘no es definido como’ clasif ‘a menos que’ características

notificación ::= ‘notificar’ mensaje ‘si’ hechos

clasif ::= identificador

sujeto ::= identificador

mensaje::= string_dec

expresión_matemática::= exp_valor

resultado::=string

exp ::= L(‘OR’ , terminoXOR)

terminoXOR::= L(‘XOR’ , terminoAND)

terminoAND::= L(‘AND’ , termino)

termino::= exp_logica |exp_conjunto | ‘(’termino‘)’

exp_valor::=L(operador_aritmetico , atomo)

atomo::= exp_elemental| ‘(’exp_valor‘)’

exp_conjunto::=L(operador_comp ,exp_valor)

exp_logica: :=‘exists’ ‘(’exp_elemental ‘,’ exp_conjunto ‘)’ | ‘empty’ ‘(’exp_conjunto‘)’

exp_elemental::=operador_conjunto ‘(’exp_conjunto ‘)’ | numero | real | booleano | string_dec | calculo ‘(’sujeto‘)’ |camino

calculo::= identificador

camino::= ‘sujeto’ O(punto camino_navegacion) | camino_navegacion

camino_navegacion ::=L(punto, elemento_nav)

elemento_navegacion::= nombre_tabla O(‘(’ macheo ‘)’)

macheo::=O(atributo op_comp)exp_valor

atributo::=identificador

nombre_tabla:= identificador

Léxico

op_comp::= ‘>’|‘<’ |‘>=’|‘<=’|‘=’|‘<>’

operador_logico :=‘and’ |‘or’ |‘xor’

operador_conjunto:= ‘sizeof’|‘sizeofdif’|‘sum’|‘sumdif’|‘avg’|‘avgdif’|‘min’|‘max’

determinante::= 'El' |'La' |'Los' |'Las' | 'Un' | 'Uno' | 'Una' | 'Cada' | 'Todos';

punto::=‘.’

dpuntos::= ‘:’

esp::= ‘ ’

div::= ‘/’

esc::= ‘-’

delim::= '\t'

meol ::= '\n'

letra: := 'a'..'z'|'A'..'Z'|'ñ'|'Ñ'|'á'|'Á'|'é'|'É'|'í'|'Í'|'ó'|'Ó'|'ú'|'Ú'

esc_char::='\\' ('n' |'t'|'"'|'\\'|'^' '@'..'_'|digito|(' ' | '\r' | '\n' | '\t' | '\f')+ '\\')

string::= (letra | digito| simbolo | esc_char | ' ' | '\t')

símbolo::='!' | '@' | '#' | '$' | '%' | '^' | '&' | '*' | '(' |')' | '_' | '-' | '+' | '=' | '{' | 'r' | '[' | ']' |':' | ';' | '<' | '>' | ',' | '.' | '?' | '~' | '/'

digito::= '0'..'9'

nueva_línea ::= delim O(meol);

ws::=# (' ' | '\n' | '\r' | '\t')

booleano::= 'true' | 'false';

identificador::= letra #(letra | digito | '_');

numero::=:N(digito);

date::= digito O (digito) div digito O (digito) div O(digito digito) digito digito
| digito O (digito) punto digito O (digito) punto O(digito digito) digito digito
| digito O (digito) esc digito O (digito) esc O(digito digito) digito digito

time::= digito O(digito) dpuntos O(digito) digito O(dpuntos O(digito) digito)

datetime::= '\"date O(N(esp) time) '\"
|'\"time '\";

string_dec::= "'string"'|'\"string'\";

real::= N(digito) O('.'N(digito)) O(('e' |'e')('+'|'-') N(digito));

operador_aritmetico ::="+" | "-" | "*" | "/ "

Semántica del LPT.

restricción ::= ::= <determinante> <sujeto> (no puede tener <características>) | (puede tener <características> solo si <hechos>

Semántica:

Establece una restricción sobre el sujeto de la regla.

Semántica de los elementos:

determinante: Es el determinante para cada sujeto, por ejemplo: Una, Uno, El, La, Cada, Todos. Según el mejor sentido en la redacción.

sujeto: Es una tabla en la Base de Datos del negocio o una clasificación de la misma.

características : Describe las características del sujeto en el negocio, tanto internas como relacionadas con otras tablas. Pueden incluir hechos con el fin de caracterizar al *sujeto*.

hechos: Hechos relativos al estado o comportamiento de la Base de Datos del negocio, incluyendo o no al sujeto.

<clasificación> ::=<determinante> <sujeto> es definido como < clasif> si <características>

Semántica:

Clasifica a un sujeto según determinadas características del mismo.

Semántica de los elementos:

clasif : Definición de un término del negocio. Típicamente define el valor de un atributo o un subconjunto de objetos en una clase existente

<notificación> ::= Notificar <mensaje> si <hechos>

Semántica:

Informa a los usuarios autorizados del negocio sobre algún conocimiento básico en tiempo real.

Semántica de los elementos:

mensaje: Mensaje de información entre comillas para usuarios autorizados del negocio.

<computo> ::= <determinante> <resultado> es calculado como <expresión matemática>
| <determinante> en <sujeto> es calculado como <expresión matemática>)
| <determinante> <resultado> en <sujeto> para <atributo> es calculado como <expresión matemática>

Semántica:

Calcula un valor determinado en el negocio, asociado o no al sujeto y su resultado es numérico.

Semántica de los elementos:

expresión matemática Definición de una expresión matemática para obtener el valor de un resultado; normalmente expresada utilizando combinaciones de términos del negocio junto a constantes disponibles.

resultado: Cualquier valor numérico, que tiene algún significado en el negocio. El resultado es usualmente el valor del atributo de un objeto del negocio.

exp ::= L(operador_logico , termino)

Semántica:

Expresa una proposición lógica entre términos booleanos en la tabla A3.1 se muestran los posibles valores del resultado en función de las expresiones que forman los argumentos.

Tabla A1.1. Posibles valores del resultado en función de los argumentos.

Operador	**Significado**	**Argumentos/Resultado**
OR	Verdadero si alguna expresión booleana es verdadera.	Booleano
AND	Verdadero si ambas expresiones booleanas son verdaderas.	Booleano
NOT	Negación de cualquier valor booleano devuelto por otro operador booleano.	Un único argumento Booleano
XOR	(a or not b) and (not a or b)	Booleano

exp_valor::=L(operador_aritmetico , atomo)

Semántica:

Los operadores aritméticos son muy útiles para las reglas de tipo cómputo, aunque pueden ser ampliamente utilizados en el contexto de otras reglas. En la tabla A4.1 se muestran los operadores aritméticos admitidos, la correspondencia de tipo entre los argumentos y el resultado.

Tabla A1.2. Correspondencia entre los argumentos de una expresión aritmética y resultado.

Operador	**Significado**	**Argumentos/Resultado**
+	Adición	Numéricos
-	Resta	Numéricos

*	Multiplicación	Numéricos
/	División	Numéricos

exp_conjunto:=L (operador_comp, exp_valor)

Semántica:

Compara entre sí los valores de dos expresiones de acuerdo a un operador. Ambas expresiones deben tener el mismo tipo. Pueden utilizarse indistintamente para comparar elementos individuales y múltiples, estos aparecen en la tabla A3.3

Tabla A1.3.· Operadores de comparación para elementos individuales y múltiples.

Operador	Significado	Argumentos	Resultado
=	Igual a	Numéricos/Booleanos/Cadenas	Booleano
>	Mayor que	Numéricos	Booleano
<	Menor que	Numéricos	Booleano
>=	Mayor igual a	Numéricos	Booleano
<=	Menor igual a	Numéricos	Booleano
<>	Diferente a	Numéricos/Booleanos/Cadenas	Booleano

exp_logica: :=’exists’ ’(’ exp_elemental ‘,’ exp_conjunto ‘)’ | ’empty’ ‘(’ exp_conjunto ‘)’

Semántica (exist):

Retorna verdadero si un elemento existe en una colección; en otro caso devuelve falso. La expresión debe representar un valor y debe ser del mismo tipo de los elementos del conjunto.

Semántica (empty):

Retorna verdadero si la colección no contiene elementos.

exp_elemental::=operador_conjunto’(’exp_conjunto’)’ | numero | real | booleano | string_dec | calculo ‘(‘sujeto’)’ |camino

Semántica (operador conjunto):

La explicación de la semántica de cada operador de conjunto aparecen en la tabla A3.4.

Tabla A1.4. Operadores sobre una colección de elementos y devuelven un valor.

Operadores	Significado
SIZEOF <colección>	Retorna cuántos elementos contiene la colección de elementos.
AVG <colección>	Retorna el promedio de una colección numérica.
SUM <colección>	Retorna la suma de una colección numérica.
MÍN <colección>	Retorna el mínimo de una colección numérica.
MÁX <colección>	Retorna el elemento máximo de la colección.
AVGDIF <colección>	Retorna el promedio de los elementos diferentes de una colección numérica.
SIZEOFDIF <colección>	Retorna cuántos elementos diferentes contiene la colección de elementos.
SUMDIF <colección>	Retorna la suma de los elementos diferentes de una colección numérica.

Ejemplos:

sizeof(sujeto.ExamenFisico.idExamenFisico) > 5

max(sujeto.Evolucion.temperatura)>38

camino::=’sujeto’ O(punto camino_navegacion) | camino_navegacion

Semántica:

Establece el medio de acceso a los atributos de tablas y posibilita la navegación. Puede terminar en una tabla o un atributo. Es admisible emplear como tabla la palabra reservada *sujeto* al inicio del camino, la cual no pertenece realmente a ninguna tabla específica, su utilización referencia al sujeto de la regla.

elemento_navegacion:= nombre_tabla O(‘(’ macheo ‘)’)

Semántica:

Representa una tabla de la base de datos. Si se especifica algún *macheo*, se restringe el valor de algún atributo perteneciente a una tabla que pertenezca al camino de navegación. En el caso que

se quiera restringir a partir de la llave primaria, puede omitirse el atributo. Esto no es posible si la llave primaria es compuesta.

Tipos de datos:

Las consideraciones sobre los tipos de datos aparecen en la tabla A3.5.

Tabla A1.5. Tipos de datos para el LPT

Tipo	Valores
Booleano	True, False
Cadena	"Esto es una cadena"
Entero	-2, -1, 0, 1, 2, 3…
Real	-0.5, 0.25, 0.5, 0.75, 1.25

ANEXO 2. **Regla de restricción en modo diferido**

Este modo de chequeo *diferido* ocurre cuando no se toman acciones inmediatamente después de ser violada la regla cuando se ejecuta una operación aislada, sino que estas reglas son consideradas como *posibles reglas infringidas.*

Funcionamiento:

1. Para cada regla de negocio tipo restricción se tiene una vista de integridad que dice si esa regla ha sido infringida y qué <sujetos> de la misma la han incumplido;
2. Para una serie de operaciones en la base de datos del negocio se extraen las reglas que han sido violadas de las posibles infractoras (Lista de Eventos) (mediante las vistas de integridad) y con estas se crea una Lista de Reglas Violadas (LRV).

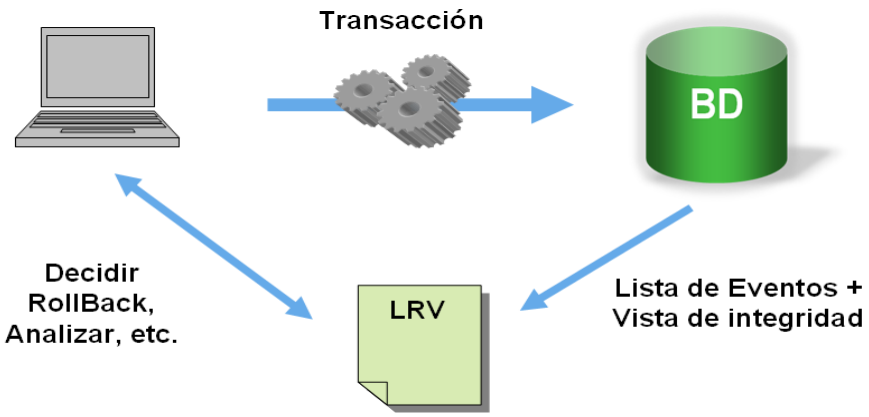

Figura A7.1. Chequeo diferido del proceso de evaluación para R.N

3. El sistema del negocio, que use el traductor de reglas de LPT a recursos SQL, debe decidir qué hacer (el usuario)
    a. si deshacer todas las operaciones o
    b. analizar las reglas que han sido violadas y de cada una cuáles son los objetos causantes de la infracción... La decisión a tomar debe tenerse desde el sistema de información ¿qué desea hacer? o realizar acciones automáticas predefinidas.

En el ejemplo de la regla:

RN#12: Un Cubículo no puede tener CANTIDAD (sujeto.Objeto.tipo = ‘cama’) <> 8

Si se desean reemplazar camas las operaciones de eliminación e inserción deben documentarse:

c. Se elimina X número de objetos tipo cama (X <= 8)
d. Se inserta una cantidad X.

Resultado: no generará violación alguna seguirán existiendo 8 camas y la transacción involucra ambas operaciones, la de eliminación y de inserción

Proceso de integridad para R.N. tipo restricción, modo diferido.

Estas políticas prohibitorias deben ser asociadas a un conjunto de operaciones y no a una única operación como tradicionalmente se llevaba a cabo [145]. La base esencial para implementar este proceso ha de ser otro de los recursos que brindan la mayoría de los gestores, definido en el estándar SQL: las transacciones [47].

Se propone que cada operación del sistema en la base de datos del negocio debe realizarse mediante un procedimiento, el cual se encarga de ejecutar esta transacción y determinar si puede ser aceptada o no.

Estructura de estos procedimientos. Para garantizar el correcto inicio del proceso de integridad en la transacción se deben tener en cuenta los siguientes pasos:

1. Realizar las operaciones necesarias del usuario.
2. Verificar las violaciones de las operaciones previamente finalizadas.
3. Análisis y decisión final sobre las infracciones cometidas.
4. Mostrar opcionalmente la información almacenada de dicha transacción.

```
CREATE PROCEDURE PTRANSAC#X ()
BEGIN
    /* Primer segmento */
    /* Segundo segmento */
    /* Tercer segmento */
    /* Cuarto segmento */
END;
```

Pasos previos:

1. Eliminar todas las instancias de la tabla RN_RESTRICCIÓN.
2. Que almacenan todas las reglas que necesitan ser exploradas luego de realizadas las operaciones.
3. comenzar la transacción de la que luego se toman determinaciones [146]. Para ello la siguiente sección de código en SQL estándar.

```
DELETE FROM RN_RESTRICCIÓN;
START TRANSACTION;
```

Ejecutar la acción o el conjunto de acciones deseados por el usuario: inserción, eliminación o actualización y conlleva a que ciertos eventos cuestionen varias reglas del negocio. Esto trae como consecuencia que se deba verificar posibles violaciones cometidas por las operaciones previamente finalizadas.

tabla RN_RESTRICCIÓN.

Para las reglas tipo restricción las posibles infractoras asociadas a las operaciones de acción que maneja el procedimiento quedan almacenadas en la tabla de RN_RESTRICCIÓN.

Segmento en un dialecto del SQL estándar (Transac-SQL)

(puede ser ejecutado en el QueryAnalyzer de SQL SERVER 5.0 [75].

1. Para cada regla de RN_RESTRICCIÓN se chequea su vista de integridad,

   en caso de violación debe almacenarse y guardarse la cantidad de <sujetos> que la infringen.

```
CREATE TABLE LRV(
            NOMBRE_RN VARCHAR NOT NULL,
                  CANT_SUJETOS INTEGER,
                        PRIMARY KEY ( NOMBRE_RN));
```

ANEXO 3. **Reglas de notificación en modo diferido**

Para recopilar la información de las notificaciones a realizar se propone una tabla auxiliar que pueda ser posteriormente consultada. Esta recibe las notificaciones de las políticas del negocio por lo que su tamaño puede aumentar vertiginosamente, debido a esto será limpiada cada un período de tiempo, definido este por los requisitos del negocio. Para una idea clara de la misma se presenta su definición utilizando el LDD [72].

```
CREATE TABLE NOTIFICACIONES(
            NOMBRE_RN CHAR (20) NOT NULL,
            FECHA DATETIME NOT NULL DEFAULT CURRENT_DATE,
            MENSAJE CHAR (100) NOT NULL,
            PRIMARY KEY ( NOMBRE_RN, FECHA));
```

Así, cada vez que se produzca un estado en el negocio que deba ser notificado se inserta en la base de datos mediante la cláusula INSERT del estándar SQL. Para la sección de código restante de la RN#7 siguiendo esta filosofía se tiene:

```
INSERT INTO NOTIFICACIONES (NOMBRE_RN, MENSAJE)
VALUES ('RN#7','Alerta Roja de la Pandemia');
```

ANEXO 4. **Modificaciones del lenguaje formal para el control de versiones de reglas**

Se selecciona la regla RN4 tomada de una BD Trasplante Renal que plantea:

Lenguaje Natural: Un paciente no puede ser mayor de 18 años y tener más de 5 exámenes físicos.

Lenguaje Técnico: Un paciente no puede tener sujeto.edad>18 and sizeof (sujeto.ExamenFisico.idExamenFisico) > 5

<idRegla>: RN4

<Versión>: 4.0

<Sujeto>: Paciente

<Característica>: sujeto.edad>18 and sizeof(sujeto.ExamenFisico.idExamenFisico) > 5

En esta regla el *sujeto* es la tabla *Paciente*, pero interviene además la tabla *ExamenFisico*.

Supongamos que esta regla se generó, compiló y se activó en la BD, por tanto van a existir tuplas que están asociadas a esta versión de regla, como se muestra en la A9.1

Con el transcurso del tiempo esta regla es modificada por la regla que plantea

Lenguaje Natural: Un paciente no puede ser mayor de 18 años y tener más de 3 exámenes físicos.

Lenguaje Técnico: Un paciente no puede tener sujeto.edad>18 and sizeof (sujeto.ExamenFisico.idExamenFisico) > 3

<idRegla>: RN4

<Versión>: 4.1

<Sujeto>: Paciente

<Característica>: sujeto.edad>18 and sizeof (sujeto.ExamenFisico.idExamenFisico) > 3

Con la activación de la regla RN4 versión 4.1 es posible que existan pacientes en la BD que:

• No violen la regla

• La violen.

En caso de existir tuplas que no entran en contradicción con la versión 4.1, se sustituye la asociación dato-regla con la versión 4.0 por la versión 4.1 en todas las tablas de relación implicadas en el camino de navegación, visto como un todo, en este caso Paciente y Exámenes Físicos.

Si existen datos que incumplen la versión 4.1, estos se conservan en la BD asociados a la versión anterior, tal y como estaba. Para ello, es necesario que los recursos que implementan la versión 4.1 de la regla RN4 no chequeen los que se conservaron asociados a la versión 4.0.

Así, la implementación de las características y los hechos deben excluir los datos implicados en cada una de las tablas del camino de navegación que no estén asociados a esa regla. Así se puede conocer con certeza cuáles datos violan la regla en cuestión. Véase figura A4.1.

Paciente

idPaciente	Edad	...
1	26	...
2	15	...
3	40	...
4	20	...

RelacionPacienteRegla

idPaciente	idregla	versión
1	RN4	4.0
2	RN4	4.0
3	RN4	4.0
4	RN4	4.1

Regla

idregla	versión	estado
RN4	4.0	0
RN4	4.1	1
...	...	...

ExamenFisico

idExamenFisico	idPaciente	...
1	1	...
2	1	...
3	2	...
4	1	...
5	3	...
6	1	...
7	3	...
8	2	...
9	4	...
10	4	...
11	4	...

RelacionExamenFisicoRegla

idExamenFisico	idregla	versión
1	RN4	4.0
2	RN4	4.0
3	RN4	4.0
4	RN4	4.0
5	RN4	4.0
6	RN4	4.0
7	RN4	4.0
8	RN4	4.0
9	RN4	4.1
10	RN4	4.1
11	RN4	4.1

Figura A4.1. Porción de la BD Trasplante Renal con las tablas de apoyo al control de versiones de RN

A continuación se muestra la generación de la regla RN4 versión 4.1, nótese que solo se chequean los pacientes y se cuentan los exámenes físicos que estén asociados a dicha regla:

Lenguaje Formal:

```
CREATE VIEW VRN4_1 AS
      SELECT * FROM Paciente sujeto
      WHERE ((SELECT a.Edad FROM Paciente a WHERE
            (sujeto.idPaciente=a.idPaciente) AND sujeto.idPaciente
                  IN (SELECT r.idPaciente FROM RelacionPacienteRegla r
            WHERE
                        r.idregla= 'RN4' AND r.version ='4.1')) > 18 AND
                        (SELECT COUNT (b.idExamenFisico)
                        FROM Paciente a, ExamenFisico b WHERE
                  (sujeto.idPaciente=a.idPaciente) AND
                        (a.idPaciente=b.idPaciente) AND
```

```
                        b.idExamenFisico IN (SELECT
                            r.idExamenFisico FROM
                            RelacionExamenFisicoRegla r
                            WHERE  r.idregla='RN4'and  r.version  ='4.1'))
                    > 3 );
CREATE TRIGGER TIRN4_1_1 ON Paciente FOR INSERT AS
     if (EXISTS (SELECT * FROM VRN4_1))
                BEGIN raiserror ('RN4',16,1);
                     rollback transaction;
                END
CREATE TRIGGER TIRN4_1_2 ON ExamenFisico FOR INSERT AS °°°°°°°°
     if (EXISTS (SELECT * FROM VRN4_1))
     BEGIN
         raiserror('RN4',16,1);
         rollback transaction;
     END
```

Para las tablas y los datos que se muestran en la figura el paciente 1 con 26 años se ha realizado 4 exámenes físicos, por tanto viola la regla 4.1. Pero como este paciente está asociado con la versión 4.0, no se tiene en cuenta.

Los pacientes 2 con 15 años y 3 con 40 años no violan la regla 4.1 porque la cantidad de exámenes que se han realizado no es superior a 3. Por tanto se debe sustituir la asociación de estos pacientes con la versión 4.0 por la regla 4.1, al igual que todos los exámenes físicos realizados a estos pacientes en la tabla de relación correspondiente.

En este caso intervienen en el camino de navegación dos tablas (Paciente y ExamenFisico); de contener más tablas se sigue el mismo procedimiento con todas las tablas y se excluyen aquellos datos que no estén relacionados con la versión de la regla que se está chequeando. De esta forma es seguro encontrar (si existen) todos los datos que violan una determinada versión de regla.

ANEXO 5. **Reflejar en las tablas de relación datos-reglas la inserción de una regla**

Una vez insertada la regla en la BD, se actualizan todas las tablas que guardan información sobre esta: en la tabla Regla se inserta una nueva tupla que contiene la información referente a esa regla como su identificador, versión y el atributo estado toma valor uno (1). Esta información de la regla se extrae del repositorio y se insertan en la tabla con la cláusula del SQL: INSERT INTO Regla VALUES...

En la(s) tabla(s) RelaciónTablaXRegla se inserta las tuplas relacionadas con la regla insertada en dependencia de la(s) tabla(s) sobre la(s) que actúe.

Si se eliminaron los datos que no satisfacen la nueva regla entonces todas las tuplas de esa(s) tabla(s) se relacionan con esa regla; pero si se conservan los datos que no la satisfacen, solo los datos que se ingresen a partir de la activación de la regla serán evaluados por ella y se registrará su asociación en las tablas de relación tupla-regla correspondientes.

En la BD se insertan los recursos con los que fue implementada y a partir de ese momento los datos que se van a regir por esa regla ya activada y las demás reglas activas.

Al inicio del trabajo con la inserción de datos, la base de datos está vacía y con las reglas iniciales activas.

Así a medida que se inserten datos estos cumplen las reglas insertadas y activas.

Para soportar el mantenimiento de una regla de negocio, se propone la actualización de las tablas RelaciónTablaXRegla añadiendo la(s) tupla(s) correspondiente(s) a esa(s) tabla(s) relacionada(s) con la nueva regla.

Si se desea conocer las tuplas sobre las cuales actúa la nueva regla, se deben realizar consultas a las tablas involucradas en el camino de navegación de esa regla. Suponiendo que se desee conocer las asociaciones con la TablaX, quedaría de la siguiente manera:

```
SELECT * FROM RelaciónTablaXRegla
WHERE idregla = RNx AND versiónregla =x.0;
```

Donde x puede tomar cualquier valor natural y representa el número de la regla.

ANEXO 6. **Repercusión en los datos por la eliminación de una regla**

- Eliminar la regla de la base de datos y conservarla en el repositorio significa:
    - Eliminar los recursos con que fue implementada.
    - Borrar todas las relaciones de esa regla con los datos
    - Eliminarla de la tabla Regla.

Cada una de las opciones se interpreta de las maneras especificadas a continuación.

Eliminar los recursos con que fue implementada

Se borran los recursos de implementación dígase disparadores y vistas, se utilizan las cláusulas del SQL: DROP TRIGGER, DROP VIEW especificando el nombre de cada recurso, cada recurso es reconocido porque se conforma teniendo en cuenta el identificador de la regla que estos implementan.

Borrar todas las relaciones de esa regla con los datos

Es necesario chequear que no se queden datos asociados a la versión de regla borrada, porque la relación no sería válida, para esto:

Buscar todas las tuplas que están relacionadas con la regla que se desea borrar.

Eliminar las tuplas correspondientes de la tabla de asociación tupla- Regla.

Eliminar las tuplas de las diferentes tablas de asociación que participan en la regla.

Cada regla se implementa mediante los recursos de BD: vista y disparadores.

La vista contiene el camino de navegación de las tablas que intervienen en la regla tipo restricción y devuelve las tuplas del sujeto que no cumple la regla.

Los disparadores se establecen sobre cada una de estas tablas.

El identificador y la versión de la regla que se desea eliminar, permite buscar en el repositorio de generación, y seleccionar de la vista correspondiente, las tablas implicadas.

Se conocen las tablas involucradas en el camino de navegación de la regla, por lo que se puede realizar la eliminación sobre las tablas de relación (*RelacionTablaXRegla*) y se eliminan todas las tuplas que están asociadas a esa versión de regla mediante la sentencia del SQL DELETE, DELETE FROM *RelacionTablaXRegla* WHERE *idregla* =RN# and *versión* = #.x;

De manera similar se borran de cada una de las tablas de relación datos-reglas, las tuplas que están asociadas a la regla RN# en su versión x, se tiene #.x.

El símbolo # significa un valor numérico entero a partir del 1, correspondiente al identificador de la regla, por ejemplo RN2. La x significa el número de la versión de dicha regla, que puede ser de

un solo nivel de profundidad como #.1 hasta #.1.1.2, etc. Por ejemplo la regla con id=RN2, versión =2.1.1

Al eliminar una regla se borran todas las asociaciones con dicha regla

Eliminar la regla de la tabla Regla

Se borrará de la tabla Regla la información correspondiente a esa regla, como se muestra a continuación:

DELETE FROM *Regla* WHERE *idregla* = RN# AND *version = #.x;*

En el **repositorio** solamente cambia el atributo *fecha_fin,* del período de activación de la regla, que automáticamente se actualiza con el valor de la fecha y hora de ese momento. Indicando que ya la regla no se encuentra activa. De esta manera se mantiene el historial de reglas en el repositorio en caso que el sistema administrador de reglas desee consultar esta información por algún motivo.

Eliminar la regla de la base de datos y del repositorio

Eliminar la regla totalmente consiste en eliminarla de la BD y de los repositorios.

Para eliminar la regla de la BD, se hace lo mismo que la opción anterior y se adiciona eliminar toda la información la regla almacenada en los repositorios.

Si la regla constituye una hoja del árbol de versiones de reglas, o sea que de ella no se deriva ninguna otra versión entonces basta con eliminarla de los repositorios. Se puede comprobar que una regla es hoja, si no existen otras reglas que contengan en el atributo padre, el identificador de la regla en cuestión.

En el caso que la regla no sea una hoja se puede podar el árbol por esa rama, como se muestra, en el Capítulo 2 epígrafe 2.7.2. Esta poda solo se puede realizar si las reglas que se derivan de ella, no estén activas en la BD.

La eliminación de una regla que no constituye una hoja del árbol, implica eliminar de la BD las asociaciones de estas reglas (la regla en cuestión y sus derivaciones) con los datos y sus filas correspondientes en la tabla *Regla*. Esto se logra de manera similar a la eliminación de una regla, con la propagación a varias reglas. Se realiza el mismo procedimiento tantas veces como reglas pertenezcan a esa rama.

Para eliminar las reglas de la tabla *Regla*, se deben anidar los identificadores y la versión de cada regla en la condición del WHERE de la sentencia DELETE.

Cuando se decide eliminar una regla, se complejiza el trabajo si existen derivaciones de las reglas, y existen relaciones con los datos se puede perder información, que luego sea necesario consultar.

No se recomienda llevar a cabo eliminaciones en el repositorio a no ser que sea tenga plena conciencia de los riesgos que se corren.

ANEXO 7**. Repercusión en los datos por la actualización de una regla de negocio**

Una vez seleccionada se le hacen las modificaciones pertinentes, se genera y compila la nueva versión de la regla.

Las acciones a realizar sobre los repositorios sobre la regla a eliminar son las siguientes:

Se eliminan de la BD los recursos con que fue implementada y en el repositorio se actualiza el atributo fecha_fin correspondiente a ese período de activación con la fecha actual.

En el repositorio se crea una nueva versión de la regla anterior, que es la regla modificada, la cual va a tener como padre a la versión anterior y se crea un período de activación en el cual se actualiza el atributo fecha_inicio con el valor de la fecha y hora actual en el repositorio.

Se tiene en cuenta que no puede existir más de una versión de la misma regla activa, ni puede existir solapamiento de fechas en diferentes versiones de una misma regla. Los recursos de implementación generados se insertan en la BD y se activa la regla modificada actualizando su atributo estado con el valor uno (1). Los recursos activos existentes de la regla antes de ser modificada, se eliminan.

La modificación de una regla, puede o no entrar en contradicción con alguna de la información contenida en la BD.

A continuación se analizan ejemplos de reglas de negocio tipo restricción extraídos de la BD Trasplante Renal.

Si tenemos activa la regla que plantea:

Un donante potencial no puede tener edad mayor que 75 años.

RN2 Donante_Potencial no puede tener

Donante_Potencial.Edad > 75

<idRegla>: RN2

<Versión>: 2.0

<Sujeto>: Donante_Potencial

<Característica>: Edad>75

Ningún paciente de los existentes en la BD, tiene edad mayor que 75 años; por tanto cumplen dicha regla, que hasta el momento está siendo aplicada en la BD.

Si se desea sustituir por la regla:

RN2 Donante_Potencial no puede tener

Donante_Potencial.Edad > 70

<idRegla>: RN2

<Versión>: 2.1

<Sujeto>: Donante_Potencial

<Característica>: Edad>70

Al cambiar la regla pudiesen existir en la BD donantes potenciales con más de 70 años, y ninguno tener más de 75.

Se debe determinar qué hacer con aquellos donantes potenciales que tienen edad con valores entre 70 y 75; pues las tuplas correspondientes a estos violan la regla que se ha modificado.

Al modificar una regla pueden existir datos que violen o no la nueva versión de la regla.

Si se decide que los datos que no violan la nueva versión de la regla a partir de ese momento se van a regir por ella, por tanto deben sustituirse las asociaciones de estas tuplas con la regla anterior, por la nueva versión. De esta manera quedarán restringidas por la nueva regla.

En caso de existir datos que entren en contradicción con la nueva versión de la regla que se desea implantar, se resuelve de manera similar al caso Insertar Regla argumentado anteriormente. Por tanto es necesario decidir que se va a hacer con esos datos:

1. Eliminarlos todos.
2. Conservarlos en la BD.
3. Seleccionar cuáles datos eliminar y el resto conservarlos en la BD.

Para cada una de estas variantes se analiza que acción ejecutar con los datos que no cumplen la nueva actualización de la regla [122].

ANEXO 8. **Repositorio primario de reglas de negocio**

```
- <Regla id="RN#9" estado="inactiva">
    <lenguajeFormal />
    <lenguajeTecnico>Un paciente no puede tener sujeto.edad >18 and
      sizeof(sujeto.ExamenFisico.idExamenFisico) > 5</lenguajeTecnico>
    <lenguajeNatural>>Un paciente no puede ser mayor de 18 años y tener
      más de 5 exámenes físicos.</lenguajeNatural>
  </Regla>
</Reglas>
```

Figura A8.1. Regla en el repositorio sin implementar.

```
- <Regla id="RN#9" tipo="restriccion">
    <ExpresionSQL>(SELECT a.Edad FROM Paciente a WHERE
      (sujeto.idPaciente=a.idPaciente)) > 18 AND (SELECT COUNT
      ( b.idExamenFisico ) FROM Paciente a , ExamenFisico b WHERE
      (sujeto.idPaciente=a.idPaciente) AND (a.idPaciente=b.idPaciente))
      > 5</ExpresionSQL>
  - <ListaEventos>
    - <Evento tabla="Paciente">
      - <TipoEvento>
          <Tipo>INSERT</Tipo>
        </TipoEvento>
      </Evento>
    - <Evento tabla="ExamenFisico">
      - <TipoEvento>
          <Tipo>INSERT</Tipo>
        </TipoEvento>
      </Evento>
    </ListaEventos>
  </Regla>
```

Figura A8.2. Parte del repositorio de generación en XML.

```
- <Regla id="RN#9" estado="activa" tipo="restriccion" ultima_fecha="2011-05-30"
    version="1.0">
  - <lenguajeFormal>
    - <Recursos>
        <Recurso tipo="TRIGGER" nombre="TIRN#9_1">CREATE TRIGGER
          TIRN#9_1 ON Paciente FOR INSERT AS if(EXISTS(SELECT * FROM
          VRN#9 ) ) BEGIN raiserror('RN#9',16,1); rollback transaction;
          END</Recurso>
        <Recurso tipo="TRIGGER" nombre="TIRN#9_2">CREATE TRIGGER
          TIRN#9_2 ON ExamenFisico FOR INSERT AS if(EXISTS(SELECT * FROM
          VRN#9 ) ) BEGIN raiserror('RN#9',16,1); rollback transaction;
          END</Recurso>
        <Recurso tipo="VIEW" nombre="VRN#9">CREATE VIEW VRN#9 AS SELECT
          * FROM Paciente sujeto WHERE ((SELECT a.Edad FROM Paciente a
          WHERE (sujeto.idPaciente=a.idPaciente)) > 18 AND (SELECT COUNT
          ( b.idExamenFisico ) FROM Paciente a , ExamenFisico b WHERE
          (sujeto.idPaciente=a.idPaciente) AND (a.idPaciente=b.idPaciente))
          > 5)</Recurso>
      </Recursos>
    </lenguajeFormal>
    <lenguajeTecnico>Un paciente no puede tener sujeto.edad >18 and sizeof
      (sujeto.ExamenFisico.idExamenFisico) > 5</lenguajeTecnico>
    <lenguajeNatural>>Un paciente no puede ser mayor de 18 años y tener más de
      5 exámenes físicos.</lenguajeNatural>
  </Regla>
```

FiguraA8. 3. Parte del repositorio de reglas.

ANEXO 9. **Descripción de clases de equivalencias**

- Clases de equivalencia que agrupa las reglas con navegación por tablas con interrelación 1: M y/o 1:1, estos caminos de navegación pueden incluir varias tablas y estas a su vez incluir un macheo.

- Clase de equivalencia donde se utilice:
  - ✓ <tabla>, reglas donde se utilicen caminos de navegación de una sola <tabla> en cualquiera de las alternativas vistas anteriormente.
- Clase de equivalencia donde se utilice:
  - ✓ <tabla>.<camino>, reglas que utilicen caminos de navegación con más de una tabla y existan interrelaciones de 1: M y/o 1:1.

De este modo como ejemplos representativos de casos se tienen:

1- *RN#13: Un Estudiante no puede tener sujeto.año >5*

2- *RN#14: Un Estudiante no puede tener sum(sujeto.Participacion_Cultura.cantidad_actos)>4*

3- *RN#15: Un Estudiante no puede tener sizeof(sujeto.Equipo.Deporte('Pelota_M').Copite.Criollos.lugar)>5*

- Clases de equivalencia correspondientes a la obtención de una colección de elementos y un elemento individual.
  - Clase que agrupan las reglas donde se intente obtener un elemento individual por alguna de las siguientes vías:
    - ✓ OP_CONJUNTO (colección), reglas donde después de obtenida una colección de elementos se realiza una operación de conjunto (sum, sizeof, etc.) sobre ella.
    - ✓ camino, reglas donde utiliza un camino de navegación que devuelve un elemento individual (en este caso los caminos de navegación no deben incluir interrelaciones 1: M).
    - ✓ CONSTANTE, reglas que utilizan una constante.
    - ✓ elemento OP_ARITMETICO elemento, reglas donde aparece una operación aritmética (+, /,*,-) entre dos elementos.
  - Clase de equivalencia para la obtención de una colección de elementos por una de las siguientes alternativas:

- ✓ colección OP_COMPARACION elemento, reglas donde aparece una operación de comparación entre un elemento individual y una colección de elementos.
- ✓ colección OP_COMPARCION colección, reglas donde aparece una operación de comparación entre dos colecciones de elementos.
- ✓ Camino, reglas donde utiliza un camino de navegación que devuelve un elemento individual (debe existir al menos una interrelación 1: M entre dos tablas involucradas en el camino de navegación).

En estas clases de equivalencia como en las que representan la obtención de una colección de elementos, se debe tener en cuenta el tipo de datos sobre el cual se está operando.

Es necesario probar que es posible obtener un elemento o una colección de elementos para cada uno de los tipos de datos almacenados en la BD del negocio.

Algunos de los casos representativos serían:

1. *RN#16: Un Estudiante puede tener sum (sujeto.Participacion_Cultura.cantidad_actos)>5*
2. *RN#17: Un Criollos no puede tener sujeto.lugar*5<25*
3. *RN#18: Un Deporte no puede tener min(sujeto.Celebra.fecha_inicio)<'18/11/2012'*

Las reglas 1 y 3 están bajo la misma clase de equivalencia (reglas donde se obtiene un elemento mediante la aplicación de un operador de conjunto sobre una colección de elementos), pero se muestran como casos separados por el tipo de datos sobre el cual se está operando. En la primera regla sobre datos tipo numérico y en la tercera sobre datos fecha.

- ➢ Clases de equivalencia para agrupar las reglas que realicen de manera similar las operaciones sobre los elementos individuales y colección de elementos para obtener un elemento booleano. Las posibles operaciones son:
  - ✓ EXIST (elemento, colección): Pregunta si existe el elemento en la colección.
  - ✓ EMPTY (colección): Pregunta si la colección está vacía.
  - ✓ elemento OP_COMPARACION elemento: Realiza una operación de comparación entre dos elementos.
  - ✓ NOT (booleano): Niega un resultado booleano.
  - ✓ booleano OP_LOGICO booleano: Realiza una operación booleana entre dos elementos booleanos.

Debido al orden lógico que se ha establecido durante la creación de estas clases de equivalencia, en este nivel solo es necesario probar el funcionamiento de los operadores de existencia (EMPTY, EXISTS), la comparación entre dos elementos individuales, los operadores booleanos y las posibles combinaciones de los mismos; cualquier vía de obtención de un elemento individual o una colección de elementos ya ha sido probada mediante las clases de equivalencias correspondientes a esos elementos.

De este modo, del grupo de reglas siguientes, las reglas 20 y 21 pertenecen a una misma clase de equivalencia (reglas de restricción en su primera alternativa donde se combinan operadores de comparación y operadores booleanos) no así en el caso de la regla 1 (reglas de restricción en su primera alternativa donde se usa el operador EMPTY):

1. *RN#19: Un Estudiante no puede tener EMPTY (sujeto.Participacion_Cultura)*
2. *RN#20: Un Criollos no puede tener sujeto.edición >2010 and sujeto.lugar >10*
3. *RN#21: Un Festival no puede tener sujeto.presupuesto > 200 and sujeto.lugar>5*

En los ejemplos mostrados solo aparecen reglas de tipo restricción pero este análisis es similar para cada uno de los tipos de reglas, donde aparece el elemento variable <características>.

Para probar la funcionalidad de los <hechos> el procedimiento es similar, aunque no es obligatorio la existencia de al menos un camino de navegación que parta de la tabla sujeto, y en el caso del <expresión matemática> se reduce a probar las posibles combinaciones de obtener un elemento individual del tipo numérico.

Entre las regla de negocio con una perspectiva de los datos pueden presentarse interdependencias. Una vez probadas las funcionalidades de cada una de las reglas por separado se deben analizar las posibles interdependencias entre ellas. Para este análisis se es posible la creación de clases de equivalencias que agrupen la interrelación de un tipo de regla con otra. Por tanto se deben chequear los siguientes casos de interdependencia, véase tabla Tabla 2.2

ANEXO 10. **Encuesta acerca del uso de las categorías de reglas con perspectiva de datos y uso del LPT**

Estimado Dr.Sc, M.Sc, Analista, especialista en bases de datos :
Esta encuesta pertenece al proceso de validación de la investigación: "Categorías de reglas de negocio desde la perspectiva de datos y su implementación automática". Usted recibe este cuestionario pues ha sido seleccionado como *Experto* por estar vinculado al desarrollo de Sistemas de información o al diseño e implementación de bases de datos. Teniendo presente su ejercicio como analista, arquitecto de SI, en el trabajo de diseño e implementación de bases de datos, consideramos que su ayuda nos sería de gran utilidad. Por tal motivo le agradecemos de antemano, que una vez que revise el material que se adjunta y que explica los propósitos de la investigación, responda la encuesta siguiente:
Nombre: _____ **Centro laboral**: _____
Cargo: _____ **Fecha**: _____ **MSc.___ Dr.Sc___ Esp.BD____ Años de experiencia_____**
Se está teniendo en cuenta que con un enfoque tradicional de desarrollo de SI, ud. necesita implementar las políticas asociadas al comportamiento de los datos dentro de la base de datos. Considere las posibilidades que da el reconocimiento de las categorías de reglas de negocio (RN) desde la perspectiva de los datos, para expresar estas reglas.

1. ¿En qué grado considera Ud. que es útil aplicar las categorías de RN desde la perspectiva de los datos, para reconocer las reglas de negocio asociados al comportamiento de los datos del negocio, que están representados en el esquema relacional de la base de datos?
   Muy alto_____ Alto____ Neutro_____ Bajo_____ Muy Bajo____
2. ¿En qué grado considera fácil reconocer las categorías desde la perspectiva de datos cuando ellas están escritas en el lenguaje natural coloquial del especialista?
   Muy alto_____ Alto____ Neutro_____ Bajo_____ Muy Bajo____
3. Al conversar con el especialista del negocio, ¿en qué medida le resulta fácil preguntarle sobre las reglas de negocio que Ud. reconoce dentro de la perspectiva de los datos?
   Muy alto_____ Alto____ Neutro_____ Bajo_____ Muy Bajo____
4. Después de reconocer las reglas desde la perspectiva de los datos, en qué medida se le facilita expresarlas en lenguaje de los patrones correspondientes a cada categoría?
   Muy alto_____ Alto____ Neutro_____ Bajo_____ Muy Bajo____
5. ¿En qué medida considera que se facilita la correspondencia entre los términos del negocio en que se expresan las reglas y los términos representados en la base de datos?
   Muy alto_____ Alto____ Neutro_____ Bajo_____ Muy Bajo____
6. Como diseñador de la base de datos considere que Ud. ha identificado las interrelaciones entre las tablas (términos) del negocio. ¿Qué grado de facilidad tiene el lenguaje de patrones técnico (LPT) para expresar las reglas escrita como patrones ?
   Muy alto_____ Alto____ Neutro_____ Bajo_____ Muy Bajo____
7. Con respecto al uso del LPT, ¿en qué medida considera que este lenguaje es completo?
   Muy alto_____ Alto____ Neutro_____ Bajo_____ Muy Bajo____
8. Teniendo en cuenta que después que escribe las reglas en LPT, se garantiza la automatización de las mismas dentro de la base de datos. ¿En qué medida considera útil reconocer las reglas con perspectiva de los datos y escribirlas en LPT ?
   Muy alto_____ Alto____ Neutro_____ Bajo_____ Muy Bajo____
9. Teniendo en cuenta el desarrollo de SI con bases de datos que hacen persistentes numerosos términos del negocio. ¿ en qué medida considera útil reconocer y escribir reglas de negocio con perspectiva de datos?
   Muy alto_____ Alto____ Neutro_____ Bajo_____ Muy Bajo____

ANEXO 11. **Encuesta acerca del conocimiento del tema por parte de los especialistas**

Estimado Dr.Sc, M.Sc, Analista, especialista o estudiante:
Como parte de la validación de la investigación: "Categorías de reglas de negocio desde la perspectiva de datos y su implementación automática", se ha seleccionado un panel de especialistas. Teniendo presente su ejercicio como analista, arquitecto de SI, en el trabajo de diseño e implementación de bases de datos, consideramos que su ayuda nos sería de gran utilidad. Por tal motivo le agradecemos de antemano, que responda la encuesta siguiente:

Nombre: _____**Centro laboral**: _____

Profesor_____ DrC____ Mr.C____ Especialista en sistemas_____ Otro cuál_____

Años de Experiencia:_____ E-mail:_____

Los estudios teóricos realizados por Ud. sobre la importancia actual del trabajo con reglas de negocio puede categorizarlos de:

Alto____ Medio_____ Bajo_____

1. Los estudios teóricos realizados por Ud sobre el diseño e implementación de bases de datos puede catalogarlos de:
   Alto____ Medio_____ Bajo_____
2. El nivel de experiencia adquirido por Ud. durante su vida profesional para el diseño de BD y trabajos con gestores de datos es:
   Alto____ Medio_____ Bajo_____
3. El conocimiento sobre que UD. posee sobre investigaciones y/o publicaciones nacionales e internacionales sobre reglas de negocio.
   Alto____ Medio_____ Bajo_____
4. Posee conocimientos generales sobre el estado de este tema de investigación.
   a. Existe un conjunto de reglas de negocio que pueden ser modeladas en el esquema lógico de la base de dato: aquellas relacionadas con la estructura del negocio que se reflejan en la estructura de los diagramas relacionales y quedan automáticamente implementadas por los gestores.
      SI____ No_____ No pero puedo intuirlo_____
   b. Existe un conjunto de reglas de negocio que no pueden ser modeladas en el esquema lógico de la BD, aquellas relacionadas con comportamiento de datos y que pudieran ser implementadas por recursos activos de bases de datos, y en este caso el programador los codifica.
      SI____ No_____ No pero puedo intuirlo_____
5. Tiene Ud. conocimiento actualizado sobre bases de datos y/o reglas de negocio adquirido en:
   Curso postgrado_____Maestría_____ Doctorado_____ Otro cuál:_____

Muchas gracias por su colaboración,
M.Sc. Martha Beatriz Boggiano Castilo.
Dr. Ramiro Alberto Pérez Vázquez
Dra. Luisa Manuela González González

ANEXO 12. **Lista de las reglas del caso de trasplante renal**

Regla Original	**Equivalencias en la BD**	Regla Equivalente
1- Un donante no puede aparecer dos veces como donante.	Un *donante* equivale a la tabla *historia clínica del donante.*	La historia clínica de un donante no puede aparecer más de una vez en las historias clínicas de los donantes
2- Un receptor no puede aparecer dos veces como receptor.	Un *receptor* equivale a su tabla *historia clínica del donante.*	La historia clínica de un receptor no puede aparecer más de una vez en las historias clínicas de los receptores.
3- Un receptor no puede ser un donante.	Un *donante* y un *receptor* equivalen a las tablas de sus respectivas historias clínicas	La historia clínica de un receptor no puede aparecer como la historia clínica de un donante.
4- Un donante no puede estar en más de una operación.	Un *donante* equivale a la *historia clínica del donante.* *Operación* equivale a la tabla *trasplantes de los donadores vivos.*	La historia clínica de un donante no puede aparecer más de una vez dentro de los trasplantes de los donadores vivos.
5- Un acto quirúrgico de un donante no puede durar más de dos horas.	Un *donante* equivale a la *historia clínica del donante.* *El acto quirúrgico de un donante* equivale a la tabla *operación del donador.*	La operación de un donador no puede durar más de dos horas.

Regla Original	**Equivalencias en la BD**	Regla Equivalente
6- Un acto quirúrgico de un receptor no puede durar más de dos horas.	*El acto quirúrgico de un receptor* equivale a la tabla *operación del receptor.*	La operación de un receptor no puede durar más de dos horas.
7- Un Trasplante Renal debe realizarse cuando el VIH del paciente fue negativo.	El *paciente* equivale a la tabla *historia clínica del receptor.* Existe una tabla donde se registra los tipos de análisis realizados a los pacientes	La historia clínica de un receptor no puede tener el resultado positivo en análisis de VIH.
8- Un Trasplante Renal debe realizarse cuando el Hepatitis B del paciente fue negativo.	El *paciente* equivale a la tabla *historia clínica del receptor.* Existe una tabla donde se registra los tipos de análisis realizados a los pacientes	La historia clínica de un receptor no puede tener el resultado positivo en análisis de Hepatitis B.
9- Un Trasplante Renal debe realizarse cuando el Hepatitis C del paciente fue negativo.	El *paciente* equivale a la tabla *historia clínica del receptor.* Existe una tabla donde se registra los tipos de análisis realizados a los pacientes	La historia clínica de un receptor no puede tener el resultado positivo en análisis de Hepatitis C.
10- Un trasplante Renal debe realizarse cuando el VIH del donante fue negativo.	El *donante* equivale a la tabla *historia clínica del donante.* Existe una tabla donde se registra los tipos de análisis realizados a los pacientes	La historia clínica de un donante no puede tener el resultado positivo en análisis de VIH.

Regla Original	**Equivalencias en la BD**	Regla Equivalente
11- Un trasplante Renal debe realizarse cuando el Hepatitis B del donante fue negativo.	El *donante* equivale a la tabla *historia clínica del donante.* Existe una tabla donde se registra los tipos de análisis realizados a los pacientes	La historia clínica de un donante no puede tener el resultado positivo en análisis de Hepatitis B.
12- Un trasplante Renal debe realizarse cuando el Hepatitis C del donante fue negativo.	El *donante* equivale a la tabla *historia clínica del donante.* Existe una tabla donde se registra los tipos de análisis realizados a los pacientes	La historia clínica de un donante no puede tener el resultado positivo en análisis de Hepatitis C.
13- Un Trasplante renal solo debe realizarse cuando los vasos del receptor son adecuados.	El *paciente* equivale a la tabla *historia clínica del receptor.* Existe una tabla donde se registra los tipos de análisis realizados a los pacientes	La historia clínica de un receptor no puede tener el resultado de los análisis de los vasos distintos de adecuados.
14- La vacunación solo debe realizarse cuando el AgHBs del paciente es positivo.	Este grupo de reglas fueron desechadas, debido a que no se acercan a los datos del negocio, ya que el término *vacunación* presente en ellas no equivale a ningún nombre de tablas o atributos de la	No se acercan a los datos del negocio
15- La vacunación solo debe realizarse cuando el HVC del paciente es positivo.		

Regla Original	**Equivalencias en la BD**	Regla Equivalente
16- El suministro de la dosis de refuerzo debe realizarse cuando la vacunación del paciente es completa.	BD.	
17- La continuación de la vacunación solo debe realizarse cuando la vacunación del paciente es incompleta.		
18- El trasplante solo debe realizarse cuando el protocolo de compatibilidad sea ejecutado.	El término *protocolo de compatibilidad* formulado en estas reglas no equivale a ningún nombre de tabla o atributo en la BD.	No se acerca a los datos del negocio
19- Un trasplante renal a un paciente con diagnóstico IRCT debe realizarse cuando se tiene un donante vivo.	Este grupo de reglas se desecharon, debido a que no se acercan a los datos del negocio, ya que el término el término diagnóstico IRCT formulado en ellas no equivale a ningún nombre de tablas o atributos de la BD.	No se acercan a los datos del negocio
20- Un trasplante renal a un paciente con diagnóstico IRCT debe realizarse cuando se tiene un donante con muerte encefálica.		

Regla Original	**Equivalencias en la BD**	Regla Equivalente
21- El Tratamiento Dialítico solo debe realizarse cuando se tiene un paciente con diagnóstico IRCT.		
22- El Examen de grupo sanguíneo al Donador solo debe realizarse cuando se mantiene la decisión de Donar Órgano.	El *término decisión de Donar Órgano* formulado en estas reglas no equivale a ningún nombre de tabla o atributo en la BD.	No se acerca a los datos del negocio
23- El Trasplante Renal solo debe realizarse cuando se tiene un paciente con diagnóstico IRCT.	El término *diagnóstico IRCT* presente en estas reglas no equivale a ningún nombre de tabla o atributo en la BD.	No se acerca a los datos del negocio
24- La evaluación radical de un posible receptor solo debe realizarse cuando se tiene historia sugerente de enfermedad vascular.	Los términos *evaluación radical e historia sugerente de enfermedad vascular* presentes en estas reglas no equivalen a ningún nombre de tabla o atributo en la BD.	No se acerca a los datos del negocio
25- Un Receptor durante la Nefrectomía debe tener la diuresis mayor o igual a 2 cc/Kg/h.	El término Nefrectomía presente en esta regla no se acercan no equivalen a ningún nombre de tabla o	No se acerca a los datos del negocio

Regla Original	**Equivalencias en la BD**	Regla Equivalente
26-Un Receptor durante la Nefrectomía debe tener la PVC entre 12 y 15 cm de H2O.	atributo en la BD.	
27-Un paciente de Consulta de Progresión en estadío 1 debe tener el FGT mayor o igual a 18 y menor o igual a 120.	Este grupo de reglas fueron desechadas, debido a que no se acercan a los datos del negocio, ya que términos como *Consulta de Progresión, estadío 1, estadío 2, estadío 3, estadío 4, estadío 5 y FGT*, formulados en ellas no equivalen a ningún nombre de tabla o atributo en la BD.	No se acercan a los datos del negocio
28-Un paciente de Consulta de Progresión en estadío 2 debe tener el FGT mayor o igual a 61 y menor o igual a 80.		
29-Un paciente de Consulta de Progresión en estadío 3 debe tener el FGT mayor o igual a 31 y menor o igual a 60.		
30-Un paciente de Consulta de Progresión en estadío 4 debe tener el FGT mayor o igual a 16 y menor o igual a 30.		
31-Un paciente de Consulta de Progresión en estadío 5 debe tener el FGT mayor o igual a 15.		

Regla Original	**Equivalencias en la BD**	Regla Equivalente
32-Un paciente debe ser reorientado al área de salud cuando su estadío es 1 o 2.		
33-Un paciente debe ser atendido en Consulta de Progresión cuando su estadío es 3 o 4.		
34-Un posible donante debe tener una edad mayor o igual a 15 y menos o igual a 55.	Un donante equivale a su La historia clínica de un donador en la BD.	La historia clínica de un donante no puede tener la edad menor que 15 o mayor que 55 años.
35-Un paciente con 6 puntos en valoración VSEN debe tener la pérdida de peso entre 1 y 2%.	Este grupo de reglas fueron desechadas, debido a que no se acercan a los datos del negocio, ya que los términos *valoración VSEN* y *pérdida de peso*, formulados en estas reglas no equivalen a ningún nombre de tabla o atributo en la BD.	No se acercan a los datos del negocio
36-Un paciente con 5 puntos en valoración VSEN debe tener la pérdida de peso entre 3 y 4%.		
37-Un paciente con 4-3 puntos en valoración VSEN debe tener la pérdida de peso entre 5 y 10%.		

Regla Original	**Equivalencias en la BD**	Regla Equivalente
38-Un paciente con 1-2 puntos en valoración VSEN debe tener la pérdida de peso mayor al 10%.		
39-Un donante vivo debe tener primera línea de consanguinidad con el paciente candidato.	un donante vivo equivale a la tabla historia clínica de un donador, el paciente a la tabla historia clínica del receptor y existe una tabla llamada parentesco que almacena lo referido a las líneas de consanguinidad	La historia clínica de un donante no puede tener asociada la de un receptor y tener línea de consanguinidad diferente de la primera.

Fuente: [139].