Diseño de Reglas Gramaticales para Transformar Documentos Técnicos Corporativos Escritos en Lenguaje Natural en Discursos Controlados

Carlos Mario Zapata Jaramillo
Departamento de Ciencias de la Computación y la Decisión
Universidad Nacional de Colombia - UNAL
Medellín, Colombia
cmzapata@unal.edu.co

Bell Manrique Losada
Programa de Ingeniería de Sistemas
Universidad de Medellín - UdeM
Medellín, Colombia
bmanrique@udem.edu.co

***Resumen*—La educción de requisitos es una de las fases más importantes en el proceso de desarrollo de software. La revisión de documentos es una de las técnicas menos usadas para educir requisitos. La literatura sugiere usarla en esta fase sobre descripciones del dominio y documentos corporativos. Aplicando esta técnica, los analistas pueden extraer conocimiento del dominio e información del negocio. En este artículo se define un marco estructural de un tipo de documento técnico corporativo, relacionado con la especificación de procedimientos. A partir de este marco, se propone un conjunto de reglas gramaticales para transformar ciertas secciones del documento técnico en un discurso en el lenguaje controlado UN-Lencep. Estas reglas se pueden aplicar posteriormente en procesos de educción de requisitos a partir de la técnica de revisión de documentos.**

***Palabras clave*—Educción de requisitos, técnicas de educción, lenguaje natural, lenguaje controlado, reglas gramaticales, patrones textuales.**

I. Introducción

La educción de requisitos (ER) constituye una fase crítica del proceso de desarrollo de software. Los errores que se cometen en esta fase son extremadamente costosos cuando hay que corregirlos en las fases finales del ciclo de vida, como la implementación y pruebas. La ER integra las tareas de educir, analizar y especificar las propiedades funcionales, de comportamiento y calidad de un sistema [1].

Las descripciones formales e informales de textos escritos en lenguaje natural se suelen usar como formas comunes de representación en ER [2]. El Lenguaje Natural (LN) es flexible, universal y de amplio uso. Durante las fases de educción y análisis de requisitos del proceso de desarrollo de software, el análisis inteligente de textos se concibe como un apoyo computacional al analista de requisitos para procesar de manera automática o semiautomática toda la información capturada [3], lo cual incluye la clasificación, priorización, determinación de calidad, traducción a especificaciones más formales y otras tareas del análisis de requisitos. El LN es también un medio útil para comunicar el contenido de políticas a empleados y personas que interactúan con una organización [4].

La mayoría de organizaciones posee información importante en sus documentos técnicos. A partir de esta información, un analista puede extraer conocimiento del dominio e información del negocio para transformarla en discursos en lenguaje controlado. Esta tarea es de gran relevancia en la captura de requisitos. La literatura sugiere usar técnicas como *Background Reading* o revisión de documentos para educir requisitos desde descripciones del dominio y guías/manuales [5]. Desde este punto de vista, el proceso de educción de requisitos se puede dividir en: selección del dominio, captura de conceptos del dominio y sus relaciones y generación de una especificación.

En este artículo se presenta una revisión y análisis de un tipo de documento técnico corporativo que se puede emplear para generar conocimiento del dominio e información útil para el proceso de ER. Se define una propuesta estructural de un documento técnico de especificación de procedimientos. A partir de esta estructura, se propone un conjunto de reglas gramaticales para transformar ciertas secciones del documento técnico (escrito en LN) en un lenguaje controlado. Esta definición puede apoyar: i) el entendimiento de la organización previo al proceso de diseño de productos software; la identificación de elementos organizacionales, del dominio y del negocio, claves para el análisis de requisitos; y, iii) la especificación de modelos y protocolos asociados con el diseño de la documentación técnica.

El resto del artículo se organiza de la siguiente manera: En la Sección II se presenta la revisión de literatura y se discuten los antecedentes. En la Sección III se propone el patrón del documento y el conjunto de reglas gramaticales. En la Sección IV se realiza el análisis del procesamiento que se propone. Finalmente, en la Sección V se presentan las conclusiones y el trabajo futuro.

II. Revisión de Literatura y Antecedentes

En esta revisión de literatura se mencionan diferentes propuestas para procesar documentos técnicos como fuentes de requisitos e información organizacional. Los diferentes enfoques son los siguientes:

A. Fuentes de requisitos en el proceso de ER

En los últimos años se intensificaron las investigaciones para identificar, considerar y analizar diferentes fuentes de posibles requisitos en el proceso de ER [6].

Tradicionalmente, la captura de requisitos se suele basar en técnicas como entrevistas y diseño de aplicaciones conjuntas (JAD) [7], entre otras técnicas enfocadas hacia el análisis de escenarios [8]. Educir requisitos desde otro tipo de fuentes, como documentos técnicos corporativos, no es común. El

proceso de educción, cuando se basa en este tipo de documentos técnicos permitiría, entre otras cosas: la comprensión y descripción de la organización y el rol que representa el sistema en el contexto [9]; el entendimiento del dominio de los participantes; el diseño de entrevistas; la aplicación de técnicas para análisis de requisitos; la generación de modelos iniciales para representar el dominio del problema.

Hoy en día, es común encontrar problemas complejos asociados con el proceso de ER [7; 10], por lo cual se requiere demasiado conocimiento experto desde diversos campos para desarrollar una aplicación de software [11]. Por esta razón, es necesario tener una visión holística del problema general, para apoyar la toma de decisiones en el proceso de ER. Sommerville *et al.* [12] consideran que otras fuentes de requisitos candidatos pueden ser una buena base para esa visión holística, aún para ambientes como el de dominio del conocimiento que propone Zhang [13]. Incluso, si el proceso de ER se lleva a cabo en un ambiente conocido y controlado, algunas características del proceso (p. ej. riesgo e incertidumbre asociados con la contextualización de requisitos, los aspectos de calidad y la ambigüedad en su definición) requieren el uso de múltiples fuentes o diversas técnicas de ER para apoyarlo adecuadamente. Algunas propuestas cercanas a este análisis buscan reducir esta complejidad con base en el uso de *frameworks* y patrones [11]. Otra aproximación se orienta hacia la transformación de conocimiento embebido en explícito, como lo abordan Stein *et al.* [14].

La literatura muestra diferentes clasificaciones de fuentes de conocimiento para el proceso de ER y diferentes técnicas a aplicar. Kotonya y Sommerville [15] proponen la siguiente clasificación: fuentes relacionadas con los seres humanos (expertos del dominio, usuarios reales, participantes, etc.) y fuentes relacionadas con artefactos (guías de uso, descripciones del dominio, leyes y regulaciones, sistemas legados, etc.). Zang [13] propone otra clasificación, al agrupar las técnicas en cuatro categorías: métodos conversacionales (p. ej. entrevistas), métodos observacionales (p. ej. etnografía), métodos analíticos (p. ej. Revisión de documentos) y métodos sintéticos (p. ej. prototipado). Byrd *et al.* [16] proponen una categorización similar, pero incluyen técnicas de adquisición para la ingeniería del conocimiento.

B. Técnica de revisión de documentos

La técnica de revisión de documentos (RD) es un método analítico que usa documentos existentes como fuente para la ER. Esta técnica se conoce en diversos contextos como *background reading*, análisis de documentos, estudios de documentación o revisión técnica.

La documentación técnica corporativa es una fuente de documentación que comprende manuales de procedimiento, reglamentaciones, políticas corporativas, reglas y estatutos de una organización. De acuerdo con Zhang [13], la técnica de RD también se usa para entender la cultura general de una organización o proyecto. En la literatura se encuentran diversos acercamientos a la aplicación de RD, como los siguientes:

- Uso de Procesamiento de Lenguaje Natural (PLN) para extraer conocimiento desde documentos existentes. Esta es una propuesta común en otras disciplinas como la ingeniería ontológica. Así lo muestra Aussenac-Gilles *et al.* [17] y Fliedl *et al.* [18], quienes usan técnicas de PLN para analizar documentos de requisitos. El uso de estas técnicas es una buena decisión cuando se tienen disponibles y en texto completo extensos documentos fuente.
- O'Shea y Exton [19] usan análisis de contenidos para extraer requisitos desde un conjunto de textos de reportes de errores, por medio de una herramienta de visualización y análisis. Este proceso usa categorías predefinidas que permiten analizar cuantitativamente los resultados, para obtener la frecuencia de uso de cada categoría. Este trabajo requiere un cierto grado de entendimiento de la organización y sus procesos, por lo cual no es útil cuando no se conoce del dominio.
- El análisis de plantillas es una técnica que no requiere una categorización fija [20]. Es un método de investigación cualitativa que se aplica para analizar material de investigación en términos textuales. Consta de un proceso iterativo e intervenido donde se crea una plantilla base de códigos a partir del análisis de un corpus. Los códigos son términos que representan posibles temas en el corpus. Esta técnica requiere alta intervención del analista, debido a que el proceso de codificación se realiza manualmente, con base en herramientas de análisis cualitativo.

C. Análisis de documentación fuente

El análisis de documentos de políticas escritas en LN se comenzó a usar recientemente en: i) descripción de procedimientos organizacionales y validación del grado de cumplimiento de reglamentaciones [21]; ii) formalización y análisis de políticas privadas organizacionales, para facilitar su interpretación y explotación, como en el proyecto Sparkle [22] a partir de un conjunto de reglas de autoría que guían el análisis.

Lévy *et al.* [23] presentan un ambiente técnico que permite construir anotaciones semánticas desde unidades textuales (p. ej. palabras, frases y párrafos) hacia unidades ontológicas (p. ej. instancias, roles y propiedades). Este trabajo proporciona una interpretación, basada en ontologías, del contenido del documento. El objetivo es la integración de políticas en sistemas de decisión empleando anotaciones semánticas y su explotación desde documentación fuente.

En análisis de requisitos escritos en LN, diferentes trabajos consideran su procesamiento, particularmente para los documentos de especificación de requisitos [24; 25]. Bajwa *et al*. [26] proponen un ambiente automatizado para traducir especificaciones de reglas de negocio escritas en inglés a un conjunto de reglas semánticas SBVR (*Semantic Business Vocabulary and Rules*). En este sistema, el dominio de negocios se representa con un vocabulario controlado y un conjunto de reglas. Un computador procesa las reglas para realizar su modelado, desarrollar análisis de consistencia o generar representaciones formales (p. ej. restricciones en OCL).

En cuanto al procesamiento de documentos técnicos de una organización, existen algunos acercamientos, por ejemplo en la gestión de documentos de políticas [21; 27]; en la descripción de procedimientos organizacionales y su análisis contrastivo respecto de reglamentaciones [28]; en reglas de autoría para políticas privadas [22].

Específicamente, para el análisis de documentos que contienen operaciones y/o procedimientos no se encontraron en la literatura aproximaciones en el contexto de la ER. Este tipo de documentos operativos se denominan comúnmente 'manuales de procedimientos', pues se diseñan para definir, desplegar, ejecutar, monitorear y mantener diferentes reglas que una organización o empresa debe cumplir y se especifican en términos de secuencias de tareas.

D. Lenguaje Controlado UN-Lencep

UN-Lencep [8] es un subconjunto del LN que se usa para especificar discursos en términos de esquemas preconceptuales. UN-Lencep se diseñó, inicialmente, para que los interesados expresaran las ideas de un dominio específico y, a partir de esta información, se desarrollara una traducción a esquemas preconceptuales. Este lenguaje controlado simplifica el proceso de obtención automática de estas especificaciones.

La sintaxis básica del lenguaje se expresa en los términos mostrados en la Tabla I. En la columna del lado izquierdo de la tabla se muestran los elementos formales expresados en el lenguaje controlado, partiendo de las expresiones del lado derecho.

TABLA I. EQUIVALENCIAS PARA LA ESPECIFICACIÓN EN UN-LENCEP

Construcción Formal	**Expresión en Lenguaje Controlado**	
A <ES> B	A es una clase de B A es un tipo de B	
A <TIENE> B	B pertenece a A B es parte de A B está incluido en A B está contenido en A B es un elemento de A B es un conjunto de A	A incluye B A contiene B A posee B A está compuesto por B A está formado por B
A <R1> B	<R1> puede ser un verbo dinámico (p.ej. A registra B, A paga B)	
C <R2> D, <SI> A <R1> B	Si A <R1> B entonces C <R2> D Dado que A <R1> B, C <R2> D Luego que A <R1> B, C <R2> D	
<SI> {COND}, <ENTONCES>	{COND} es una condición expresada en términos de conceptos. <R1> y <R2> son verbos dinámicos. <SINO> es opcional, por ejemplo: Si M es más grande que 100, entonces A registra B	

III. PROCESAMIENTO DEL DOCUMENTO TÉCNICO

El uso de técnicas de extracción y procesamiento de información desde documentos de texto no estructurados, permite identificar información estructurada. En esta sección se presenta un acercamiento hacia esta identificación, a partir del marco estructural de un documento técnico corporativo.

Para tal fin, dicha aproximación se basa en el método de Bernardos y Aguado [29], quienes proponen el diseño de textos desde sus unidades estructurales o patrones, como base para su correcta generación y procesamiento.

A. Marco Estructural del documento

En esta sección se define un marco estructural del tipo de documento técnico denominado manual de procedimientos. El marco propuesto especifica qué parte del documento fuente se puede procesar en cada sección. En esta versión inicial estructural se restringen las secciones del documento a secuencias de apartados que corresponden a unidades textuales (un conjunto de oraciones, palabras y frases). Cada unidad textual se considera como relevante para el dominio bajo estudio.

Las secciones que se tienen en cuenta en este marco estructural preliminar, son nombre, objetivo, reglas de operación y descripción narrativa, así:

- El *nombre del procedimiento* corresponde a la designación dada al procedimiento
- El *objetivo del procedimiento* muestra los resultados que se pueden obtener al llevar a cabo las actividades y tareas que describe el procedimiento.
- Las *reglas de operación del procedimiento* se usan para expresar una restricción sobre cómo la solución o paso del procedimiento se espera que se comporte. Existen varias clases de restricciones, algunas de las cuales se orientan a delimitar las condiciones sobre las acciones que desarrollan los agentes.
- La *descripción narrativa del procedimiento* expresa lo que un actor es responsable de llevar a cabo en términos de acciones. Contiene el conjunto de pasos a cumplir para lograr los objetivos.

B. Caso de prueba

Siguiendo la idea de Brodie *et al*. [22], la cual se fundamenta en que es necesario analizar el lenguaje natural para entender políticas, reglas y procedimientos que, por defecto, son tareas complejas, es preciso definir reglas para transformar esta información en un lenguaje controlado. Buscando esta meta, se inició con el análisis de reglas de procedimientos que se colectaron desde porciones de texto extraídas del modelo SPEED [30]. SPEED (siglas en francés de procedimientos escritos que se siguen en entornos dinámicos) es un modelo que describe el uso de procedimientos operativos en situaciones dinámicas. El modelo comprende nueve etapas, que simulan la forma como un piloto reacciona frente a una situación específica en una operación aérea. Ellas son:

1. Detección de condiciones de activación. En esta etapa el piloto detecta de forma inicial las causas de la situación actual.
2. Elaboración de diagnóstico. El piloto ejecuta un diagnóstico de la situación para determinar el tipo o naturaleza del procedimiento a usar.
3. Determinación de la necesidad de un procedimiento. En esta etapa, para usar un procedimiento, el piloto evalúa si la información es útil y estima si la necesita.
4. Evaluación y búsqueda de información procedimental. Si el piloto necesita asistencia, él busca un procedimiento y debe ser capaz de encontrar el indicado.

Las anteriores secciones se extrajeron de SPEED, porque son consistentes con las secciones definidas previamente para un documento de procedimientos. Los apartados textuales se extrajeron del procedimiento '*Elaboration of diagnosis'* desde SPEED, y son los que aparecen a continuación:

> *Procedure name: 'Elaboration of diagnosis'*
>
> *Operating rules: To use a procedure, the pilot assesses whether the information is useful and estimates if he needs to use the procedure.*
>
> *Narrative description: If the pilot has time, he will try to understand the dysfunction before applying the procedure. If the pilot is in an emergency situation, he first applies the procedure, but while applying it, tries to understand the dysfunction by comparing his own action plan with that presented to him.*

La definición de estas secciones radica en la identificación de categorías de análisis y especificación de reglas de escritura de cada una. Alinear un tipo de procedimiento con una categoría de éstas hace más fácil entender la intención de cada paso del procedimiento y distinguirlo de los otros. Por ejemplo, el uso de "*shall*" o "*must*" denota una acción obligatoria, mientras que "*shall be able to*" denota una acción condicional. Para realizar este tipo de análisis el paso siguiente será el análisis gramatical, para la identificación de reglas de transformación.

Con ánimos de ejemplificación, se propuso un discurso para las tres secciones extraídas del procedimiento ejemplo. Este discurso se basa en las reglas derivadas del lenguaje UN-Lencep (ver apartado A) y en las tareas de análisis de términos y verbos de acción.

Se propone un discurso que resulta de analizar el procedimiento completo y su contexto de uso, con base en lo que describe Brito [30] en su modelo. Dentro de esta descripción, la expresión '*If the pilot has time'* se relaciona con el estado de 'criticidad' de la situación de emergencia. De este escenario se puede deducir que, si la situación de emergencia es demasiado crítica, el piloto no tendría que revisar la información y tendrá que aplicar el procedimiento inmediatamente. Si la 'situación de emergencia' no es demasiado crítica, el piloto primero analiza la 'situación', 'compara' el plan y sigue los pasos restantes. A partir de este análisis, la tercera parte del discurso propuesto, correspondiente a la descripción narrativa, se podría reescribir como sigue: *If the emergency situation is not critical—or status=non-critical—, the pilot analyzes the emergency situation*. Así, el discurso generado es el siguiente:

i. The pilot 'assesses' whether the status of information is useful.

ii.The pilot 'applies' the procedure when the information is needed.

iii. If the emergency situation is not critical, the pilot analyzes the emergency situation.

iv. If the pilot 'recognizes' an emergency situation, then the pilot applies the procedure. When the pilot applies the procedure, then the pilot 'analyzes' the situation and 'compares' the action_plan.

C. Reglas gramaticales para la transformación

Una regla gramatical permite identificar información útil del procedimiento a partir de su aplicación sobre el texto que lo contenga. Para este caso preliminar, se usó ANTLR (*ANother Tool for Language Recognition*) como marco de trabajo para analizar un grupo de sintaxis derivadas desde documentos técnicos. Esta herramienta se dirige hacia los lenguajes de programación, donde las sintaxis son menos flexibles que el LN, dándole flexibilidad al análisis de las unidades textuales extraídas desde documentos.

Se construyó un prototipo en ANTLR que toma como entrada una gramática definida, es decir, una descripción precisa del lenguaje con acciones semánticas definidas, y luego genera un lenguaje controlado. A partir de esta gramática, se busca información extraída de cada unidad textual de los procedimientos, que se pueda procesar, analizar y finalmente escribir en UN-Lencep, como lenguaje de salida.

Las reglas propuestas constan de estructuras gramaticales que describen el comportamiento de las situaciones y acciones expresadas en la descripción del procedimiento: las situaciones son condiciones sobre cuándo se tomará una acción; las acciones son comandos o instrucciones. Las reglas gramaticales propuestas se clasifican por categorías y se describen con la sintaxis que obliga ANTLR, como aparece a continuación:

- Categoría estructural, que implica el uso de verbos estructurales o relaciones permanentes entre conceptos ("*to be*", "*to have*", and "*to only have*"). Esta categoría define la Regla 1 y se implementa como sigue.

$$\begin{array}{c} noun + HAVE + noun \\ noun + IS + noun \end{array} \quad (1)$$

```
static_rule returns [String value] : exp  = ruleIS {$value = $exp.value;} | exp1 = ruleHAVE {$value = $exp1.value;}
    ruleIS returns [String value] : exp1=noun {$value = $exp1.value;}
    'IS ' exp2=noun {$value = $exp1.value + " is " + $exp2.value;}
```

- Categoría dinámica, la cual implica el uso de verbos dinámicos o relaciones temporales entre conceptos (verbos tales como "*to assess*", "*to apply*", "*to analyze*", etc.). Esta categoría reconoce las relaciones dinámicas en diferentes formas. Como un verbo es la realización de una acción, es importante identificar la acción incluso cuando no se identifica el actor que la ejecuta o el objeto sobre el cual recae. Por esta razón, se define la Regla 2 para identificar las posibles ocurrencias:

$$\begin{array}{c} noun + VERB + noun \\ noun + VERB \\ VERB + noun \end{array} \quad (2)$$

```
dynamic_rule returns [String value] : exp1=substantive {$value = $exp1.value;}
    (exp2 = verbs_function exp3=noun {$value = $exp1.value + $exp2.value + $exp3.value;})
    | exp2 = verbs_function {$value = $exp2.value;}
    ( exp6=noun {$value = $exp2.value + $exp6.value;})
    | exp7=substantive {$value = $exp7.value;}
    ( exp8= ws exp9 = verbs_function {$value = $exp7.value + $exp8.text + $exp9.value;} )
    | exp10= verbs_function
```

- Categoría implicativa. Representa relaciones causa-efecto entre relaciones dinámicas. Sus usos pueden ser: como enlace entre relaciones dinámicas, o como enlace entre un condicional y una relación dinámica. Para identificar el inicio de una implicación se identificaron las palabras reservadas: *if* y *when.* Para identificar la forma de una implicación, se usan las palabras reservadas *then*, *now* y *before*, como se muestra en la Regla 3, donde R es una relación.

$$\begin{array}{c} If + R + then + R \\ noun + VERB \\ VERB + noun \end{array} \quad (3)$$

```
implication_rule returns [String value]
    : exp= condition exp1 = relations {$value = "if " + $exp1.value;}
    (' THEN ' exp2 = relations {$value = "if " + $exp1.value + " then " + $exp2.value;}
    |' NOW ' exp3 = relations {$value = "if " + $exp1.value + " then " + $exp3.value;}
    |' BEFORE' exp4 =relations {$value = "if" + $exp1.value +"then" + $exp4.value;})?
```

- Categoría de reglas compuestas. Indican la unión entre diferentes reglas básicas (1, 2 o 3) en la misma frase.

$$Regla\ de\ Relaciones \quad (4)$$

relations returns [String value] : exp = static_rule {$value = $exp.value;} | exp1 = dynamic_rule {$value = $exp1.value;}
| exp2 = ruleadjetive {$value = $exp2.value;}

Regla de Relación + Implicación (5)

rule_unlencep returns [String value] : exp = relations {$value =$exp.value;}
| exp2 = implication_rule {$value = $exp2.value;}

Regla de Conjunción (6)

relation_conj returns [String value] : (exp= rule_unlencep) + {$value = $exp.value;}
((exp2= conj)? (exp3= rule_unlencep)+)? {$value = $exp.value + '\n' + $exp3.value;}

El discurso original, previo a la aplicación de las reglas, es:

To use a procedure, the pilot assesses whether the information is useful and estimates if he needs to use the procedure. If the pilot has time, he will try to understand the dysfunction before applying the procedure. If the pilot is in an emergency situation, he first applies the procedure, but while applying it, tries to understand the dysfunction by comparing his own action plan with that presented to him.

El discurso generado en UN-Lencep tiene la apariencia que se muestra en la Tabla II.

TABLA II. DISCURSO EN UN-LENCEP DESDE PROCEDIMIENTO ORIGINAL

Procesamiento del texto		
Texto de Entrada	***No.frase extraída***	***Texto de Salida***
to use a procedure the pilot assesses whether the information is useful	*1.* *2.* *3* *4.*	*to use procedure* *pilot assesses* *information have state* *state is useful*
estimates if he needs to use the procedure	*5.*	*if he needs procedure*
if the pilot has time he try to understand the dysfunction before applying the procedure	*6.* *7.* *8.*	*if pilot has time* *he tries dysfunction* *applying procedure*
if the pilot is an emergency he applies the procedure	*9.* *10.*	*if pilot is emergency* *He applies procedure*
but while applying it tries to understand the dysfunction	*11.*	*tries dysfunction*
by comparing his own action plan with that presented to him	*12.*	--

IV. ANÁLISIS DEL PROCESAMIENTO

De acuerdo a la salida del prototipo en ANTLR, las siguientes son las mayores debilidades en el discurso de salida, sobre las cuales se está trabajando:

- Ausencia del actor que dirige las acciones, pues en muchos casos no es explícito quién está usando el procedimiento (frase 1).
- Uso del término '*if*' sin la consecuente acción '*then'*(frase 5).
- Existencia dentro de una frase de dos o más verbos juntos (p.ej. '*try to understand*'). El prototipo solo toma el primer verbo (frases 5, 6, 11).
- La expresión '*the pilot has time'* (frase 6) es una afirmación sin sentido en este contexto, pues no es una propiedad medible de forma práctica.
- Cuando en una frase condicional no se incluye el término '*if*' (frases 5, 6), la frase de salida no se toma como una implicación, por la falta de la palabra reservada.
- La frase '*pilot is an emergency'* (frase 9) no es adecuada, debido a que se interpreta como si una emergencia fuera un posible valor del *piloto* o como si *piloto* fuera un tipo de *emergencia*, no como una situación del ambiente que el piloto percibe. Por esta razón, la salida no es clara ni precisa.
- El prototipo no procesa el término '*but'* antes de una expresión condicional como *while* (frase 11).
- No se incluyeron las preposiciones dentro de la gramática, como en el caso de '*by'* (frase 12). Por esta razón, el prototipo no reconoce este carácter y por ende, no genera resultados.

A partir del análisis de los resultados preliminares, es necesario que los textos que se procesen con el prototipo sigan ciertas condiciones sintácticas, para que la máquina las pueda interpretar. Para visualizar las diferencias en las frases de salida cuando el discurso de entrada cumple con ciertas 'mejores prácticas de sintaxis', en la Tabla III se muestra el discurso final de salida, a partir del siguiente discurso de entrada modificado:

The pilot 'assesses' whether the status of information is useful. The pilot 'applies' the procedure when the information is needed. If the emergency situation is not critical, the pilot analyzes the emergency situation. If the pilot 'recognizes' an emergency situation, then the pilot applies the procedure. When the pilot applies the procedure, then the pilot 'analyzes' the situation and 'compares' the action_plan.

TABLA III. DISCURSO EN UN-LENCEP DESDE EL PROCEDIMIENTO MODIFICADO

Procesamiento del texto		
Texto de Entrada	***No.frase extraída***	***Texto de Salida***
the pilot assesses whether the information have status is useful	*1.* *2.*	*pilot assesses* *information have status*
The pilot applies the procedure when the information is needed	*3.* *4.*	*information have status* *if status is needed then pilot applies procedure*
If the emergency situation is not critical, the pilot analyzes the emergency situation	*5.* *6.*	*emergency_situation have status* *if status is non-critical then pilot analyzes emergency_situation*
if the pilot recognize an emergency then the pilot applies the procedure	*7.*	*if pilot recognize emergency then pilot applies procedure*
When the pilot applies the procedure then the pilot analyzes the situation and compares the action_plan	*8.* *9.*	*if pilot applies procedure then pilot analyzes situation* *compares action_plan*

Finalmente, se puede afirmar que para el análisis de este tipo de documentos técnicos es importante trabajar de forma inicial en aspectos de su escritura, redacción y estructura. En principio, son necesarios los siguientes aspectos que se identificaron:

- Acciones con agente o actor inexistente.
- Acciones escritas en voz pasiva
- Uso de términos no estándar
- Acciones que usan nombres diferentes para referirse a la misma entidad
- Uso de dos expresiones de acción continuas, sin mucha relevancia en la oración. En la mayoría de los casos existe un mejor término para expresar lo mismo.

V. Conclusiones y trabajo futuro

En este artículo se parte del uso de técnicas de ER, basadas en la revisión de documentación. El valor de uso de documentos técnicos corporativos en esta área es incremental.

Muchos estudios intentan reducir los problemas asociados con PLN, limitando el alcance del lenguaje. Algunos usan sub-lenguajes, pero estos no constituyen realmente LN. A partir de este trabajo y algunos antecedentes, se propone delimitar las gramáticas para considerar solo un subconjunto del LN usado para escribir documentos técnicos corporativos. Sin embargo, es difícil no usar un lenguaje restringido para simplificar la tarea de traducción a un LC, lo que impone diversas restricciones sobre la libertad de expresión de los participantes en el proceso.

En el análisis de procedimientos, se lograron hallazgos relacionados con suposiciones que el lector o analista del documento deben hacer para poder ejecutarlo o tomar decisiones respecto de la situación que se presente. Esto implica que el texto por sí solo no permite tomar este tipo de decisiones y consume tiempo el analizar situaciones como: significados comunes para diferentes términos sin previa aclaración; acciones sin agente o actor que las dirija; uso de verbos compuestos con una ambigüedad muy alta.

El procesamiento de documentos técnicos (p. ej. un manual de procedimientos), específicamente un conjunto de procedimientos como en el caso de prueba, tiene diferentes ventajas, entre las que se cuentan: reducción de sobrecarga de trabajo en la interpretación, minimización de los posibles errores humanos con su uso; estandarización del desempeño de operaciones humanas. Por estas razones, la gestión de esta clase de documentos tiene una utilidad incremental y se propone tenerlos en cuenta como fuente de información útil para el proceso de educción de requisitos en la ingeniería de software.

La investigación actual y futura se centra en modificar las gramáticas base del prototipo, buscando mejorar las frases generadas en el lenguaje controlado sin hacer suposiciones. También, se pretende incluir otras secciones del documento de procedimientos. Se planea trabajar con más conjuntos de procedimientos organizacionales para direccionar temas contextuales, así como:

- Reconocer adverbios y sus reglas asociadas, así como las comas dentro de las frases y su papel en las implicaciones.
- Identificar desde una lista de verbos, cuáles de ellos son los más importantes en una frase.

Agradecimentos

Las Vicerrectorías de Investigación de la *Universidad de Medellín* y la *Universidad Nacional de Colombia*, financiaron este trabajo en el marco del proyecto de investigación: "MÉTODO DE TRANSFORMACIÓN DE LENGUAJE NATURAL A LENGUAJE CONTROLADO PARA LA OBTENCIÓN DE REQUISITOS, A PARTIR DE DOCUMENTACIÓN TÉCNICA".

Referencias

[1] Castro-Herrera, C., Duan, C., Cleland-Huang, J. & Mobasher, B. (2008). Using data mining and recommender systems to facilitate large-scale, open, and inclusive requirements elicitation processes. In: Proceedings of the 2008 16th IEEE international requirements engineering conference (RE'08), Barcelona, pp 165–168. doi:10.1109/RE.2008.47

[2] Mich, L., Franch, M. & Inverardi, P. N. (2004). Market research for requirements analysis using linguistic tools. Requirements Eng, Vol. 9 (1). pp. 40–56.

[3] Casamayor, A., Godoy, D. & Campo, M. (2011). Mining textual requirements to assist architectural software design: a state of the art review. In: Artificial Intelligence Rev. Springer Science+Business Media B.V. DOI 10.1007/s10462-011-9237-7

[4] Brodie, C.A, Karat, C. M. & Karat, J. (2006). An empirical study of natural language parsing of privacy policy rules using the SPARCLE policy workbench. In Proceedings of the second symposium on Usable privacy and security (SOUPS '06). ACM, New York, NY, USA, pp. 8-19..

[5] Sitou, W. & Spanfelner, B. (2007). Towards Requirements Engineering for Context Adaptive Systems. Proceeding COMPSAC '07 Proceedings of the 31st Annual International Computer Software and Applications Conference - Volume 02 IEEE Computer Society Washington, DC, USA.

[6] Islam, S., Knauss, E., Jürjens, J. & Schneider, K. (2010). Eliciting security requirements and tracing them to design: an integration of Common Criteria, heuristics, and UMLsec. Siv Hilde Houmb Requirements Engineering, 2010, Volume 15, Number 1, Pages 63-93

[7] Christel, M. & Kang, K. (1992). Issues in Requirements elicitation. Technical Report CMU/SEI-92-TR-012 ESC-TR-92-012. USA: Software Engineering Institute.

[8] Zapata, C.M., Gelbukh, A. & Arango, F. (2006). Pre-conceptual Schema: A Conceptual-Graph-Like Knowledge Representation for Requirements Elicitation. In: MICAI 2006, LNAI 4293, pp. 27-37. Springer-Verlag.

[9] Leite, J. (1987). A survey on requirements analysis. Advanced Software Engineering Project Technical Report RTP071. USA: Department of Information and Computer Science, University of California.

[10] Coulin, Ch. & Sahraoui, Abd-El-Kader. (2008). A Meta-Model Based Guided Approach to Collaborative Requirements Elicitation. SE-081010, Sahraoui.

[11] Kleppe, A. (2009). Software Language Engineering: Creating Domain-Specific Languages Using Metamodels (1 Ed.). Addison-Wesley Professional. Pearson Education, ISBN 0321553454.

[12] Sommerville, I., Sawyer, P. & Viller, S. (1998). Viewpoints for requirements elicitation: a practical approach. Computing Department, Lancaster University, Lancaster, UK, pp. 74-81.

[13] Zhang, Z. (2007). Effective Requirements Development – A Comparison of Requirements Elicitation techniques. In INSPIRE2007, Tampere, Finland.

[14] Stein, S., Lauer, Y. & El-Kharbili, M. (2009) Using Template Analysis as Background Reading Technique for Requirements Elicitation. Available online: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.145.7424

[15] Kotonya, G., & Sommerville, I. (2004). Requirements Engineering: Processes and Techniques. Chichester, UK: Wiley Editors.

[16] Byrd, T. A., Cossick, K. L., & Zmud, R. W. (1992). A Synthesis of Research on Requirements Analysis and Knowledge Acquisition Techniques. MIS Quarterly, 16(1), 117–139.

[17] Aussenac-Gilles, N., Biébow, B., & Szulman. S. (2000). Revisiting Ontology Design: A Method Based on Corpus Analysis. Knowledge Engineering and Knowledge Management.

Methods, Models, and Tools: 12th International Conference (EKAW), 1937, 27–66.

[18] Fliedl, G., Kop, C., Mayr, H.C., Salbrechter, A., Vohringer, J., Weber, G., & Winkler, C. (2007). Deriving static and dynamic concepts from software requirements using sophisticated tagging. Data & Knowledge Engineering, 61(3), 433–448.

[19] O'Shea, P., & Exton, C. (2004). The Application of Content Analysis to Programmer Mailing Lists as a Requirements Method for a Software Visualization Tool. 12th International Workshop on Software Technology Practice (STEP), 30– 39.

[20] King, N. (1998). Template Analysis. In Symon, G. and Cassel, C. (Eds.), Qualitative Methods and Analysis in Organizational Research: A Practical Guide (118–134). London, UK: Sage Publications.

[21] Dinesh, N., Joshi, A., Lee, I. & Sokolski, O. (2008). Reasoning about conditions and exceptions to laws in regulatory conformance checking. In: DEON 08, pp. 16. Available Online: http://repository.upenn.edu/cispapers/371

[22] Brodie, C.A, Karat, C. M. & Karat, J. (2006). An empirical study of natural language parsing of privacy policy rules using the SPARCLE policy workbench. In Proceedings of the second symposium on Usable privacy and security (SOUPS '06). ACM, New York, NY, USA, pp. 8-19.

[23] Levy, F., Guisse, A.,Nazarenko, A., Omrane, N. & Szulman, S. (2010). An Environment for the Joint Management of Written Policies and Business Rules. 22nd IEEE International Conference on Tools with Artificial Intelligence (ICTAI). IEEE Computer Society, Vol. 2, pp. 142-149.

[24] Ambriola, V. & Gervasi, V. (2006). On the systematic analysis of natural language requirements with circe. In: Automated Software Engineering, Vol. 13, pp. 107–167.

[25] Santiago, V., Vijaykumar, N. L. & S. da Silva, J. D. (2009). Natural language requirements: Automating model-based testing and analysis of defects. In: IX Workshop do Curso de Computao Aplicada. LIT/INPE. Available Online: http://www.lac.inpe.br/cap/arquivos/pdf/P29.pdf

[26] Bajwa, I. S., Lee, M. G. & Bordbar, B. (2011). SBVR Business Rules Generation from Natural Language S pecification. Association for the Advancement of Artificial Intelligence. Artificial Intelligence for Business Agility — Papers from the AAAI 2011 Spring Symposium (SS-11-03)

[27] Dinesh, N., Joshi, A., Lee, I. & Sokolski, O. (2007). Logic-based regulatory conformance checking," in 14th Monterey Workshop. ScholarlyCommons@Penn, 2007. Available Online: http://repository.upenn.edu/cispapers/392

[28] Dinesh, N., Joshi, A., Lee, I. & Webber, B. (2006). Extracting formal specifications from natural language regulatory documents. In: ICoS-5, Buxton, England.

[29] Bernardos, M. & Aguado, G. (2001). A new approach in building a Corpus for Natural Language Generation Systems. In: CICLing 2001, LNCS 2004, pp. 216-225. Springer-Verlag.

[30] De Brito, G. (2002). Towards a model for the study of written procedure following in dynamic environments. In: Reliability Engineering and System Safety 75, pp. 233-244. Elsevier Publisher.

Carlos Mario Zapata Jaramillo. Recibió su título de Ingeniero Civil en 1991, una Especialización en Gerencia de Sistemas Informáticos en 1999, una Maestría en Ingeniería de Sistemas en 2002 y un Doctorado en Ingeniería con énfasis en Sistemas en 2007. Todos los títulos los recibió en la Universidad Nacional de Colombia. Es Profesor Asociado del Departamento de Ciencias de la Computación y de la Decisión de la Universidad Nacional de Colombia, sede Medellín. Sus áreas de interés son: ingeniería de software, ingeniería de requisitos, lingüística computacional y estrategias didácticas para la enseñanza de la ingeniería.

Bell Manrique Losada. Recibió su título de pregrado en Ingeniería de Sistemas en la Universidad Distrital FJC - Universidad de la Amazonia en 2003, su título de MSc. in Ingeniería de Sistemas en la Universidad Nacional de Colombia en 2006 y actualmente está cursando su Ph.D. en Ingeniería -Sistemas e Informática en la Universidad Nacional de Colombia. Es profesora asistente en la Universidad de Medellín e investigadora activa del Grupo de Investigación ARKADIUS. Sus intereses de investigación se orientan hacia las áreas de Educción de Requisitos en Ingeniería de Software, Diseño de Software y Patrones de modelado, Educación en Ingeniería de Software y Procesamiento de Lenguaje Natural para el desarrollo de Software