HERRAMIENTA SOFTWARE PARA LA GENERACIÓN AUTOMÁTICA DE TEXTOS EN INGLÉS: UN ACERCAMIENTO A LA GENERACIÓN DE LENGUAJE NATURAL

CLAYDERMAN JOSUÉ ROJAS JIMENEZ
WILLIAM LEONARDO GARCÍA RUEDA

ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA

UNIVERSIDAD INDUSTRIAL SANTANDER

Bucaramanga Octubre de 2007

HERRAMIENTA SOFTWARE PARA LA GENERACIÓN AUTOMÁTICA DE TEXTOS EN INGLÉS: UN ACERCAMIENTO A LA GENERACIÓN DE LENGUAJE NATURAL

CLAYDERMAN JOSUÉ ROJAS JIMENEZ
WILLIAM LEONARDO GARCÍA RUEDA

Trabajo de grado para optar por el título de Ingeniero de Sistemas

Director
MPe. HENRY ARGUELLO FUENTES
Co-Director
MSc. ELKIM FELIPE ROA FUENTES

ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
UNIVERSIDAD INDUSTRIAL SANTANDER
Bucaramanga Octubre de 2007

DEDICATORIA

A mi padre Leonardo García Salazar y a mi madre Olga Rueda Miranda por haber sido la razón fundamental de haber terminado mis estudios en ingeniería de sistemas, por medio de su apoyo, motivación y ejemplo de vida me dieron las fuerzas para salir adelante.

A las familias Berrio Izasa y Bautista Otálora, por su desinteresada colaboración, consejo y servicio durante los últimos cuatro años , siempre los recordaré.

A mis tíos Vidal, Elizabeth, Maria Edith y Nubia por toda la fuerza espiritual y moral que me han brindado, sus orientaciones y alegrías me dan la fortaleza para continuar cada día con felicidad en la elaboración de mi trabajo.

Finalmente a Dios por darme la gracia de vivir y permitirme aportar mi grano en el desarrollo de la ciencia.

WILLIAM LEONARDO GARCÍA RUEDA

DEDICATORIA

A Dios, a mi madre Delcy María Jiménez Cuellar y a mi padre Justo Elías Rojas Meléndez, a mi hermana Margareth, a mi familia en general que siempre fue mi fuente de inspiración para obtener este triunfo. Sólo ellos han estado de manera incondicional a través de todos los años de mi existencia, con sus sabios consejos y sus regaños, siempre buscando lo mejor para mi.

A las personas que me acompañaron en los últimos años y fueron apoyo incondicional. De manera muy especial a Doña Inés, quien considero parte importante de mi vida, mi segunda madre.

A todos mis amigos y compañneros de la Universidad Industrial de Santander. También a los que me soportaron, también a los que estuvieron en las buenas y en las malas

CLAYDERMAN JOSUÉ ROJAS JIMÉNEZ

AGRADECIMIENTOS

Nuestros mas sinceros agradecimientos a Elkim Felipe Roa y al profesor Henry Arguello por la oportunidad que nos brindaron su compromiso y dedicación durante el desarrollo de este trabajo. Fueron pilares en nuestro conocimiento personal y profesional.

A la Universidad Industrial de Santander, escuela para nuestro desarrollo personal y hogar que nos acogió para comenzar este proceso.

A los integrantes del grupo CIDIC por su valiosa colaboración, quienes contribuyeron a establecer las pautas de este proyecto.

A todos aquellos que siempre creyeron en nosotros y que de alguna manera aportaron para poder alcanzar nuestros sueños.

RESUMEN

TITULO:
HERRAMIENTA SOFTWARE PARA LA GENERACIÓN AUTOMÁTICA DE TEXTOS EN INGLÉS: UN ACERCAMIENTO A LA GENERACIÓN DE LENGUAJE NATURAL.[1]

AUTORES:
CLAYDERMAN JOSUÉ ROJAS JIMÉNEZ, WILLIAM LEONARDO GARCÍA RUEDA [2]

PALABRAS CLAVES:
texto, Generación de Lenguaje Natural, Teoría de la estructura retórica, corpus, similitud semántica.

DESCRIPCIÓN:
La redacción de textos en libros, tesis, artículos y otros documentos implica esfuerzo y tiempo para los autores, originando la necesidad de herramientas computacionales que colaboren o asistan esta labor. En Generación de Lenguaje Natural (GLN) se han desarrollado sistemas como el *KPML* [33] y el *MIAKT* [1] y técnicas en procesos intermedios de generación textual como la teoría de la estructura retórica (*RST*) y los *corpus* lingüísticos, no sólo en la obtención automática de textos , sino en la presentación de información de modo compresible a las personas a partir de datos en formato de tablas, plantillas y entre otras formas.

En este trabajo, el problema de la GLN es tratado de forma aproximada mediante la implementación de un prototipo computacional que produce textos para artículos técnicos en inglés como ayuda en la redacción de esta clase de reportes. Para ello, inicialmente se proporcionan los conceptos fundamentales de la GLN, luego se presenta una revisión de las herramientas relacionadas con el proceso, tomadas en la implementación del primer prototipo software que ejecuta un procedimiento básico de obtención de nuevo texto.

Los resultados de las pruebas a este programa determinaron la necesidad de construir un *corpus* lingüístico con apoyo humano y el diseño de dos estrategias de producción de frases una basada en árboles de búsqueda y otra en cálculo de probabilidades, empleando la similitud entre oraciones, lo cuál fue llevado en conjunto en el desarrollo del sistema *DISGEN*. El resultado del trabajo se considera un primer aporte a la Generación de Lenguaje Natural en el ámbito de la Universidad Industrial de Santander y deja abierta una referencia para la realización de proyectos relacionados a nivel local.

[1] Proyecto de investigación.
[2] Facultad de Ingenierías Físico Mecánicas, Escuela de Ingeniería de Sistemas e Informática, Fernando Ruíz.

ABSTRACT

TITLE:
SOFTWARE TOOL FOR AUTOMATED ENGLISH TEXT GENERATION: AN APPROACH TO NATURAL LANGUAGE GENERATION.[1]

AUTHORS:
CLAYDERMAN JOSUÉ ROJAS JIMÉNEZ, WILLIAM LEONARDO GARCÍA RUEDA [2]

KEY WORDS:
text, Natural Language Generation, Rethorical Structure Theory, corpus, semantic similarity.

DESCRIPTION:
Drafting texts in books, thesis, papers and other documents represents effort and time to authors, making necessary computational systems for work or attend that. In Natural Language Generation (NLG), has been developed systems such as *KPML* [33] and *MIAKT* [1], and techniques for generation intermediate process such as rethorical structure theory (*RST*) and linguistics *corpus*, not only for automatic texts obtaining, also in data presentation in a form understandable for people from data in tables, templates and other forms.

In this work, NLG problem is treated in an aproximated way by implementing a prototype computer that produces texts for papers in English language as aid in the drafting of such writings. For this, initially is provided NLG fundamental concepts, then is presented a review of tools associated with process, taken for implementation of a first software prototype that execute a basic procedure in new text obtaining.

Results of this program identified the need of building a linguistics *corpus* with human help and design of two sentences production strategies, first based on trees search and second in probabilities calculations, using similarity between sentences brought together in *DISGEN* system development. Results of this work is considered as a first contribution to Natural Language Generation in Industrial University of Santander domain and leaves open a reference for the realization of projects related in local level.

[1] Investigation Project.
[2] Faculty of Mechanical Engineering's Physical, System and Computer Science School, Fernando Ruiz.

Contenido general

Índice de tablas

Índice de figuras

Capítulo 1

Introducción

Una de las formas empleadas para transmitir conocimientos en el área académica ha sido la escritura de artículos, libros, tesis, entre otros. La edición de contenidos para esta clase de documentos implica esfuerzo, gasto de tiempo y dinero para los autores, lo cual ha suscitado la necesidad de desarrollar sistemas que apoyen la automatización de las tareas relacionadas con este proceso, y de esta forma ayudar a los autores en una redacción de texto de mayor calidad en un menor tiempo. Existen investigaciones y aplicaciones en el escenario científico mundial, principalmente en generación de lenguaje natural, como *MIAKT* [1], *IDtension* [2], entre otros.

El área de la generación de lenguaje natural GLN, en computación lingüística está dedicada al estudio de las técnicas empleadas por los sistemas de software que cuentan con la capacidad de producir texto automáticamente en lenguaje natural [3][1]. Las herramientas computacionales basadas en GLN tienen la capacidad de generar automáticamente textos de diversos tipos empleando conocimiento relacionado con un idioma y un tema específico. Algunos de los tipos de texto a generar pueden ser documentos, reportes, discursos, entre otros.

Los sistemas de GLN pueden ser diseñados e implementados con base en diversas técnicas. No obstante, existe un consenso acerca de las tareas básicas necesarias para llevar a cabo la producción y salida de texto partiendo de una entrada de datos, así como de la arquitectura global del sistema. Por tanto, en el diseño de un generador automático de texto, inicialmente deben tomarse decisiones acerca de la organización global del sistema y la selección de las herramientas a utilizar en términos de las tareas y los módulos que lo componen.

En este trabajo, el problema de la GLN es abordado mediante la implementación de una herramienta computacional con capacidad de generar automáticamente un determinado tipo de párrafo en inglés. Los resultados son evaluados a partir de pruebas realizadas a párrafos de introducción formados por seis oraciones, los cuales hacen parte de artículos técnicos relacionados con el tema de diseño de amplificadores operacionales en el área de ingeniería electrónica. La razón de esta limitación se debe principalmente a la cantidad de texto y el tema empleado. Estos factores pueden ocasionar problemas de coherencia al momento de hacer la generación. Sin embargo, se dejan las bases teóricas para que futuras investigaciones mejoren la herramienta y superen estas dificultades. Con el fin de realizar un trabajo conforme a las herramientas y metodologías existentes en la literatura, se presenta un estudio sobre los fundamentos conceptuales de la GLN y las herramientas *software* de código abierto utilizadas en la implementación de este proyecto.

[1] Entiéndase por lenguaje natural, el lenguaje creado por los seres humanos a través de su historia.

1.1. Objetivos

1.1.1. Objetivo general

- Implementar una herramienta computacional capaz de construir prototipos de texto para artículos técnicos en inglés, como apoyo al proceso de edición del documento, empleando técnicas de generación de lenguaje natural.

1.1.2. Objetivos específicos

- Estudiar y proporcionar los fundamentos conceptuales para la generación de lenguaje natural basados en los adelantos de la lingüística computacional.
- Seleccionar y aplicar procesos de adaptación de *software* a herramientas computacionales de código abierto existentes relacionadas con las tareas de generación de lenguaje natural, conforme a las necesidades específicas para la edición de texto en artículos técnicos en inglés.
- Implementar un prototipo de *software* que valide una estrategia planteada en la generación automática de párrafos para introducción y conclusiones en inglés, en el área específica del diseño de circuitos integrados.

1.2. Conceptos básicos sobre la generación de lenguaje natural

La generación de lenguaje natural se define como la rama del procesamiento de lenguaje natural[2] especializada en el estudio y desarrollo de sistemas computacionales con capacidad para producir texto. Estos sistemas toman generalmente como entrada, alguna representación no lingüística de información (tablas, plantillas) relacionada con el objetivo de la comunicación y emplean conocimiento sobre un idioma y un campo de aplicación específico, para producir automáticamente documentos, reportes, explicaciones, mensajes de ayuda, entre otros tipos de texto [3]. Bajo esta definición, se puede caracterizar a los sistemas de GLN mediante el esquema de la figura 1.1.

1.2.1. La GLN desde una perspectiva de investigación

La GLN como subárea del PLN, puede ser vista como un subcampo tanto de las ciencias computacionales, como de las ciencias del conocimiento. Estas ciencias ofrecen una importante perspectiva a varios problemas de fondo en GLN, los cuales se mencionan a continuación.

Las **ciencias computacionales** son un área importante en la actualidad, al encargarse de plantear problemas básicos respecto a la interacción hombre-máquina. La forma en que los computadores interactúan con las personas, la mejor manera de transmitir información desde una máquina hasta un ser humano y el modelado del comportamiento lingüístico esperado por las personas de un computador, son algunos de los planteamientos formulados por las ciencias computacionales en relación a la GLN.

Entre las **ciencias del conocimiento** se destacan la lingüística, la filosofía y la psicología. Éstas generan cuestionamientos relacionados con la manera de formalizar las restricciones del lenguaje a nivel semántico, sintáctico, psicológico y práctico, así como el papel del contexto en

[2]PLN o *NLP*, de las siglas en inglés: *Natural Language Processing.*

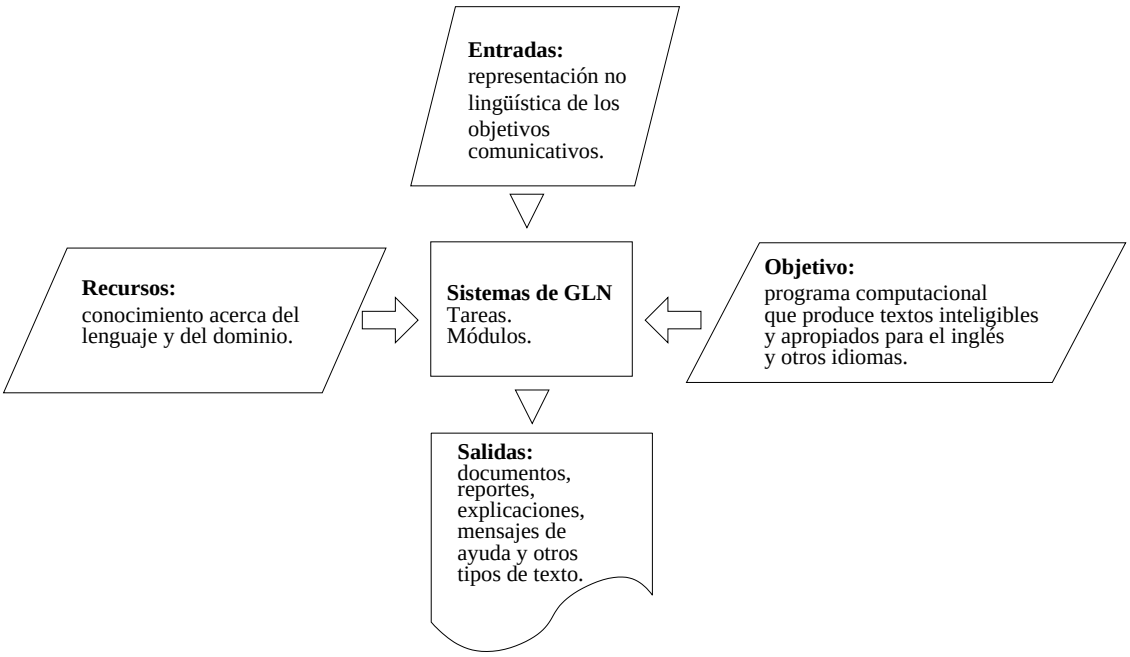


Figura 1.1: *Esquema general de los sistemas de generación de lenguaje natural.*

la elección del lenguaje que se constituye como legible o apropiado en determinadas situaciones [3].

Por otro lado, la investigación en la GLN pone en evidencia aspectos más amplios acerca del modelamiento y simulación de la inteligencia humana, uno de los principales objetivos de la **inteligencia artificial** (IA). Lo anterior implica el conocimiento de modelos de dominio y razonamientos asociados, requeridos para traducir información representada computacionalmente a lenguaje natural, con el vocabulario y la estructura apropiada para las personas.

1.2.2. La GLN desde una perspectiva de aplicación

En la actualidad, los sistemas de GLN también son empleados en la presentación de información y/o en la automatización parcial o total de la producción de documentos. La gran mayoría de los datos se almacenan en formatos electrónicos, necesitando ser codificados por los sistemas computacionales para su posterior interpretación por parte del humano. Por ejemplo, las bases de datos de horarios de las aerolíneas, las hojas de cálculo de cuentas, las bases de conocimiento de sistemas expertos y las simulaciones de sistemas físicos, basadas en datos tabulados, son manejables para un computador pero no siempre comprensibles para una persona. Esto indica la necesidad de sistemas con la capacidad de mostrar y resumir los aspectos más importantes de estos datos, facilitando así su compresión a usuarios con poca experiencia.

La producción automática de documentos es importante porque las personas pierden mucho tiempo en la edición de documentos, en situaciones donde esta actividad no es su principal responsabilidad. Un médico por ejemplo, puede tardar una significativa parte del día en la escritura de cartas de diagnóstico, licencias de maternidad y otros documentos relacionados con su rutina. Del mismo modo, un programador de computadores puede gastar varias horas escribiendo textos (documentación de código, descripción de la lógica de un programa, reportes de avances y otros), siendo similar al tiempo empleado en la realización del *software*.

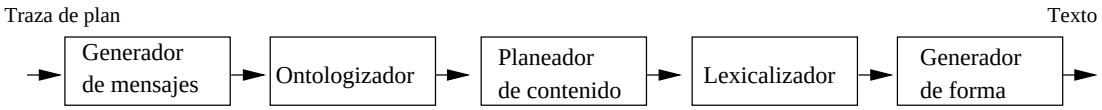


Figura 1.2: *Arquitectura del sistema PlanDoc.*

Finalmente, un académico local perderá algunas semanas en la elaboración de un artículo sobre una de sus investigaciones, cuando desea publicar su trabajo. Sumado a esto, es probable requerir la edición en otro idioma como el inglés, por lo que, seguramente el tiempo de edición aumentará al no ser éste su idioma nativo. Las herramientas que dan soporte a las actividades mencionadas producen documentos para elevar considerablemente la productividad de las personas.

En el desarrollo de un sistema de GLN, se debe considerar si éste operará generando textos sin necesidad de intervención humana o si producirá prototipos de texto para ser modificados posteriormente por un autor humano. Esta distinción en la forma de operar el sistema es semejante a lo planteado por los sistemas expertos, en dónde lo más indicado es estimar la presencia de una persona en el proceso, especialmente en situaciones en las cuales un error del sistema podrá tener consecuencias negativas. Para el caso de las aplicaciones de GLN, la presencia de una persona en el proceso de producción de texto tiende a desaparecer, porque el objetivo de estas herramientas es generar la salida de manera automática. Sin embargo, no es apropiado crear textos con la calidad y el contenido requerido sin la intervención humana al no existir métodos que evaluén los resultados obtenidos [5].

1.2.3. Algunos ejemplos de sistemas GLN

Para presentar de manera más concreta la perspectiva de aplicación de los sistemas de GLN, a continuación se hace una breve descripción de algunos ejemplos de sistemas generadores de texto, específicamente los siguientes: *PlanDoc*, *FoG*, *Drafter* y *ProtoPropp* [6].

PlanDoc es un generador de reportes para la documentación de las operaciones de planeación de una red de teléfonos. Su arquitectura está organizada secuencialmente mediante los siguientes cinco módulos (figura 1.2): el generador de mensajes, el ontologizador, el planeador de contenido, el lexicalizador y el generador de forma. El sistema toma como entrada un registro de seguimientos, producto del uso de los ingenieros con una herramienta de planeación de redes llamada *PLAN* y produce un resumen de la interacción. La entrada se transforma en mensajes mediante el generador de mensajes y continúa hacia el ontologizador, el cual enriquece cada mensaje con conocimiento semántico relacionado con el dominio de discurso de *PLAN*. El planeador de contenido determina la información que debe aparecer en el texto y lo organiza mediante la combinación de los mensajes. El lexicalizador selecciona las palabras necesarias en los mensajes y finalmente, el generador de forma produce una oración con base en los mensajes recibidos y sus léxicos. El sistema *FoG* genera textos de pronóstico del tiempo en inglés y francés, a partir de simulaciones numéricas producidas por una supercomputadora, además contienen anotaciones hechas por un meteorólogo. El primer pronóstico del estado del tiempo con uso realizada por *FoG* fue publicado en febrero de 1992.

El sistema *Drafter* es un generador de texto instruccional para manuales de *software*. Este desarrollo se basa en análisis de requerimientos de usuario, donde el autor especifica el conocimiento de los procedimientos necesarios para usar el *software* y luego genera prototipos de manuales en inglés, francés y posteriormente en italiano. El sistema comprende dos módulos: una herramienta para el desarrollador y un generador, el cual se subdivide en tres

módulos: el planeador de texto, el planeador de oraciones y el realizador de forma. El autor crea un modelo del texto a generar, mediante la herramienta del desarrollador. El modelo llega al planeador de texto en forma de mensaje y se convierte luego en un plan retórico. El planeador de oraciones realiza la estructura de las oraciones (tiempos, combinación de frases) especificada en comandos *SPL(Sentence Planning Language)* para cada idioma. A partir de estas especificaciones, el realizador de forma produce las oraciones.

Finalmente se presenta el módulo de generación de texto de *ProtoPropp* [6]. Éste es un sistema para la generación de historias asentado sobre una ontología en el cual se incluyen las características de los conceptos y las relaciones existentes entre ellos. La estructura de datos generada es un plan de trama, donde se describe un esqueleto del argumento en términos de elementos de la ontología manejada por el sistema: personajes y sus atributos, escenarios del cuento, eventos acontecidos entre otros. A partir de ahí el subsistema de GLN obtiene la representación textual de la historia.

1.2.4. Reseña histórica

La generación de lenguaje natural ha estado continuamente en la atención de los investigadores desde sus orígenes en los años 70. Dado el gran potencial y el rango de intereses implicados, la GLN ha ido evolucionando con un crecimiento acelerado durante las últimas dos décadas.

En los inicios de los años 70 se implementaron dos referencias de generación textual, una basada en redes semánticas [8] y otra en gramática sistémica. La generación a partir de redes semánticas, toma como entrada un conjunto de nodos que representan significados de palabras y relaciones entre dichos significados, y produce el contenido de la red a través de un mecanismo de transición de gramática. Una aplicación de la gramática sistémica se implementó por Davey en un programa capaz de describir cualquier juego de *tic-tac-toe* (triqui) a través de una secuencia de frases [9].

Los sistemas de GLN empiezan a producir textos con múltiples oraciones para finales de los años 70. Los generadores seguían estrategias para decidir el material a incluir y cómo organizarlo, como el sistema basado en el algoritmo Hill-Climbing *KDS*, o sistemas ejecutados de acuerdo a la organización de sus dominios semánticos [10][3].

Los inicios de los años 80 fueron de interés e investigación para la generación de lenguaje, enfocándose hacia la creación de diversos sistemas para manejar varios aspectos en el proceso de generación [12], como los actos lingüísticos, la planeación y los componentes tácticos de los sistemas de GLN. El proceso de generación es visto en función de un componente estratégico, determinación de contenido y un componente táctico, planes de texto. De estos sistemas, se destacan: *TEXT, KAMP, MUMBLE y PENMAN*.

En la mitad de los años 80, se creó la teoría de la estructura retórica (*RST Rethorical Structure Theory*), la cual tiene su origen en la generación automática de textos. Un equipo de investigadores del Instituto de Ciencias de la Información en la Universidad del Sur de California, trabajaba en la autoría de textos mediante computadora[4]. Hacia 1983 parte del equipo (Bill Mann, Sandy Thompson y Christian Matthiessen) observó la ausencia de teorías acerca de la estructura o la función del discurso que aportaran suficientes detalles para programar un generador automático de textos. En respuesta, se concibió la *RST* con base en estudios realizados a un conjunto de textos editados provenientes de diversas fuentes. La *RST* consiste básicamente en la identificación de las relaciones de coherencia existentes entre los

[3] *TALESPIN*, un programa escritor de historias.
[4] *Information Sciences Institute*, visitar *Natural Language Group* http://www.isi.edu.

segmentos individuales de un texto, a partir de las cuales se origina una estructura jerárquica de dichas relaciones, centrándose en la descripción, y no en procedimientos textuales de creación o comprensión. En la actualidad, la *RST* tiene validez y consideración dentro de la lingüística independientemente de sus aplicaciones computacionales [13].

En 1983 surgió el desarrollo de la gramática de árbol adyancente [14] como una alternativa a los estándares formales sintácticos, los cuales se usaban en análisis teóricos del lenguaje (Sigla en inglés *TAG Tree Adjoining Grammar*). La propiedad clave de los *TAGs* es permitir la división de las características lingüísticas recursivamente, a partir de un dominio de dependencias. McDonald y Pustejovsky [15] en 1985 retomaron la idea de los *TAGs* como una teoría aplicable en la generación de lenguaje.

A mediados del año 1986, el trabajo de Eduard Hovy se destacó por motivar la perspectiva pragmática y aplicable de la planeación de texto, con *PAULINE* [22][5]. En general, este sistema abarcó simultáneamente un amplio rango de problemas tratados de manera particular por diferentes programas de generación de lenguaje. En esa época los investigadores en GLN reconocen un problema al momento de editar texto, al no identificar la forma adecuada de especificar la sintaxis gramatical de un texto de tal forma que el texto generado sea aproximadamente igual al escrito por un experto. Este problema se conoce como expresiones de referencia. Una expresión de referencia contiene información acerca de una unidad de contenido para identificarlo unívocamente en el contexto del discurso actual, evitando redundancias [3].

En los 90 aparecieron las bases de conocimiento con técnicas aplicables de GLN como *WIP*, la cual utiliza documentos multimodales[6] para comunicar información [23][7]. También se desarrollaron otros tipos de herramientas como generadores de oración, uno de ellos *Genesys* [24][8], los lexicones, los analizadores sintácticos, los *corpus* anotados, los analizadores de lenguajes naturales, los desambiguadores entre otros.

Si bien la GLN lleva más de treinta años, los adelantos y aplicaciones son aún limitados. Esto se evidencia en la existencia de desacuerdos en algunas cuestiones básicas, y los pocos resultados experimentales no permiten realizar evaluaciones, y tampoco desarrollar metodologías para medir la eficiencia de los textos generados. Los trabajos actuales han sido revisados, mejorados y aplicados en diferentes idiomas. Nuevas investigaciones en esta rama sugieren aplicaciones multilenguaje y nuevas arquitecturas GLN con el fin de obtener eficiencia en los procesos de generación de texto. Los investigadores asumen *a priori* la fuerte dependencia de sus trabajos al campo de aplicación, al no existir las suficientes teorías formales que soporten la independencia de dominio para la GLN. Sin embargo, hay un significativo número de nuevos sistemas, con tendencia a ser más robustos y generales.

1.3. Estado del arte

En [25] se presenta, el número de proyectos relacionados con generación de lenguaje natural por periodos, desde 1961 hasta el 2004 (tabla 1.1). La lista al momento de consultar, registra 360 proyectos en GLN. Sin embargo la lista no es exhaustiva, porque no todos los proyectos tienen sitios *web* o referencias bibliográficas vigentes.

De la tabla 1.1 se puede observar que desde la aparición de la GLN en la década de los sesenta hasta el año 2000, ha habido un aumento gradual en el número de aplicaciones. Entre 1960 y 1989 el interés e investigación se enfocaron en aspectos específicos de los procesos

[5] Planning and Uttering Language In Natural Environments.
[6] Un documento es multimodal cuando puede bosquejar información combinando texto y gráficos
[7] *Knowledge-Based Presentation of Information.*
[8] *GENErates SYStemically.*

Tabla 1.1: *Proyectos en GLN clasificados por periodos.*

Periodo	**Proyectos**
1960-1969	7
1970-1979	26
1980-1989	82
1990-1994	102
1995-1999	103
2000-2004	38

Tabla 1.2: *Trabajos reportados en sistemas de GLN.*

Ref.	**Nombre**	**Dominio de aplicación**	**Idiomas**	**Año**
[27]	*TALE-SPIN*	Historias	Inglés	1977
[28]	*TEXT*	Investigación naval	Inglés	1980
[30]	*PENMAN*	Propósito general	Inglés	1981
[31]	*PAULINE*	Propósito general	Inglés	1986
[32]	*FoG*	Clima	Inglés, Francés	1989
[33]	*KPML*	Propósito general	Múltiples	1993
[34]	*PLANDOC*	Telecomunicaciones	Inglés	1994
[35]	*Dada Engine*	Propósito general	Independiente	1996
[36]	STOP	Medicina	Inglés	1997
[37]	*DRAFTER II*	Instrucciones	Inglés, Francés, Italiano	1998
[38]	*XTRAGEN*	Propósito general	Independiente	2002
[39]	*SKILLSUM*	Habilidades para lectura	Inglés	2003
[40]	*SCIGEN*	Ciencias computacionales	Inglés	2005
[2]	*IDtension*	Historias	Inglés	2007

propios de la GLN, creando las bases conceptuales y prácticas que definieron las arquitecturas de los sistemas actuales. Luego, durante la década de 1990, los resultados obtenidos a partir de los periodos anteriores motivaron la elaboración de trabajos multilenguaje, esto es importante si se tiene en cuenta que la GLN nació en EEUU e Inglaterra. Desde el año 2000, la GLN se ha integrado con las actuales plataformas de desarrollo de *software*, dando así lugar a aplicaciones integradas con multimedia (gráficas, animaciones, sonido, etc) y la creación de sistemas de generación a partir de bases de conocimiento (reglas para describir los mecanismos de razonamiento que permiten resolver problemas). Sin embargo, la investigación en la GLN ha sido aún más importante que la creación de sistemas, cuyos esfuerzos se enfocaron en el desarrollo de técnicas y formalismos para resolver problemas específicos en las tareas propias de la GLN. Estos aspectos cubren áreas como la coherencia del lenguaje generado, la evaluación de la calidad de los sistemas de GLN y los recursos lingüísticos.

En la tabla 1.2 se presentan las características de algunos proyectos representativos en GLN, la mayoría de los cuales están referenciados en [26].

Todos los proyectos referenciados en GLN generan texto en inglés, nueve de los cuales lo hacen exclusivamente en este idioma. *KPML (Komet-Penman Multilingual Development Environment)* es el único generador de oraciones multilenguaje, ofrece soporte para generar texto en 11 idiomas[9]. Existen además dos generadores basados en gramáticas, *Dada Engine*

[9] Al momento de consultar, el KPML registra soporte para generación en inglés, holandés, alemán, francés,

y XtraGen, con estructuras de soporte para varios lenguajes, los cuales dan flexibilidad a la generación de texto.

Los dominios de aplicación de los proyectos referenciados son variables, cinco son de propósito general y para el resto, el campo de aplicación se menciona sólo una vez. Pero en general, se evidencia la alta dependencia de los proyectos con su campo de aplicación, ya que inicialmente las entradas a los sistemas son restringidas a un dominio específico. Por esta razón también tienden a ser de corto periodo de utilización.

En resumen, este proyecto plantea la implementación de un generador de determinados tipos de párrafos en inglés, aprovechando el conocimiento y las herramientas existentes en el campo de la GLN. Las pruebas del software se realizaron para párrafos de introducción, conformados por seis oraciones. La razón principal por la cual se limita la herramienta se debe a los problemas de coherencia ocasionados por la cantidad de texto a generar y el área de aplicación utilizado, lo anterior no asegura la eficacia en los resultados, sólo intenta realizar una aproximación al problema. Sin embargo, se dejan las bases teóricas para futuras investigaciones, esperando mejoras en la herramienta y el tratamiento de estas dificultades. Específicamente se aplica al área del diseño de circuitos integrados sobre *OPAMPs* por la necesidad de mejorar la edición del contenido de los artículos producidos en el grupo de diseño de circuitos integrados CIDIC.

1.4. Organización del documento

El capítulo 2 describe la arquitectura de un sistema de GLN, explicando el comportamiento de las entradas, la salida, las tareas, los módulos y las clases de arquitecturas más utilizadas en generación de lenguaje natural, mencionando paralelamente su relación con este trabajo. En el capítulo 3 se hace una revisión a las herramientas del área de estudio y se muestra un primer prototipo resaltando los problemas y limitaciones de generar texto automáticamente. En el capítulo 4 se presenta la propuesta de generación de texto basado en las limitaciones según lo expuesto en el capítulo anterior. En el capítulo 5 se muestran las pruebas del sistema en términos de escenarios, con el respectivo análisis de sus resultados, las conclusiones del trabajo y las recomendaciones pertinentes.

japonés, checo, ruso, búlgaro, español, griego y chino.

Capítulo 2

Generación de texto

En el capítulo anterior se presentó el marco contextual relacionado con la generación de lenguaje natural, abordando aspectos como su definición, áreas de estudio, ejemplos de aplicación, reseña histórica y estado del arte. Esta revisión resalta la necesidad de desarrollar sistemas de GLN para mejorar la presentación de información textual y la producción de documentos en diferentes disciplinas. Así mismo, se destacan las limitaciones de dichos sistemas respecto al idioma y al campo de aplicación.

Buscando presentar con más detalle el proceso de la GLN, a continuación, se describen los diferentes conceptos vinculados con la estructura de un generador de texto, sus entradas, salida y tareas, así como los módulos y las tipologías más utilizadas. En la medida que se mencionan los anteriores tópicos se hace una analogía de los mismos con el trabajo realizado. Finalmente, se hace un resumen del capítulo con el objetivo de sentar los fundamentos necesarios para dar una compresión general de la forma como se genera texto automáticamente.

2.1. Entradas, salida y tareas

Las entradas a un sistema de GLN constan de información representada en tablas, informes y datos estadísticos relacionados con un propósito a comunicar, junto con varias fuentes de conocimiento. Algunos de estos propósitos pueden ser informar, persuadir, describir, ayudar a un usuario y/o suministrarle información. Formalmente, la entrada a un sistema de GLN está conformada por 4 elementos: la fuente de conocimiento, el objetivo a comunicar, el modelo del usuario y el modelo del discurso.

La fuente de conocimiento es toda la información necesaria para hacer funcionar un sistema de generación de texto. Esta información se puede representar en bases de datos relacionales, tablas, redes semánticas, entre otros. En este trabajo se usan las bases de datos relacionales, los *corpus* de texto y los artículos técnicos como fuentes de conocimiento.

El objetivo a comunicar es la finalidad del texto que se va a generar, como describir, explicar, o analizar. Pueden además incluirse restricciones sobre el texto resultante, por ejemplo, cierto número de oraciones, párrafos o páginas. El objetivo del generador de prueba de este trabajo es realizar el resumen de un artículo técnico de amplificadores operacionales integrados mediante un párrafo de introducción de seis oraciones.

En el modelo de usuario se tienen presentes aspectos relacionados con las necesidades propias de la persona, la cual utiliza la herramienta. En el diseño del sistema se identificaron 5 roles de usuario: invitado, experto, revisor, editor y administrador. El modelo del discurso registra los acontecimientos comunicados a través del sistema. Sin embargo, cabe anotar que

los modelos del usuario y del discurso no siempre están presentes explícitamente en los sistemas de GLN.

Los recursos lingüísticos no se especifican en el conjunto de entrada de los sistemas de GLN, estos recursos constituyen los componentes internos del sistema. El uso de diccionarios electrónicos *(WordNet y Moby-Thesaurus)* y el etiquetador *PoS* apoyan al humano en la construcción del *corpus* y se constituyen en los recursos empleados por el sistema, cabe destacar que son herramientas utilizadas para el inglés pues es el idioma sobre el cual se ha realizado importantes desarrollos respecto a la generación de lenguaje natural. No obstante, es factible concebir un sistema en donde los recursos lingüísticos formen entradas independientes, por ejemplo, un sistema multilingüe produce un documento en varios idiomas, pero requiere de una gramática y un lexicón del idioma en particular.

La salida de un sistema de GLN es, por supuesto, texto en lenguaje natural comprendido por un ser humano. Como se mencionó, las pruebas hechas al sistema generan texto en párrafos de seis oraciones que resumen una publicación técnica sobre *OPAMP*. Los trabajos en el área de la GLN sólo se interesan en el texto producido (cómo sartas de caracteres) sin importarles los detalles de su formato. El sistema propuesto genera texto plano, aunque opcionalmente podría agregársele un leve formateado con etiquetas LaTeX[1].

Un sistema de GLN relaciona algunos datos de entrada a una salida de texto por medio de un proceso. No obstante, como muchos procesos computacionales, puede descomponerse en varias subtareas debidamente definidas, delimitadas y distinguibles. Aunque existe controversia respecto a cuáles deberían ser estas subtareas, en la comunidad de la GLN, se ha llegado a un acuerdo que determina siete actividades básicas a llevar a cabo para ir desde los datos de entrada hasta la salida. Las siete tareas básicas para un sistema de GLN son las siguientes: determinación de contenido, planeación del discurso, agregación de oraciones, lexicalización, generación de expresiones de referencia, realización lingüística y realización estructural.

Esto no significa que un sistema de GLN necesite siete módulos, pues muchos sistemas más completos poseen módulos capaces de desempeñar varias tareas simultáneamente. En este trabajo se hace énfasis en la agregación de oraciones por medio del empleo de un *corpus* de texto, dos estrategias de generación y similitud de frases. Sin embargo, para tener una idea más amplia se describe de manera general el proceso.

Determinación de contenido: consiste en seleccionar el tipo de información a ser comunicada en el texto. Para el caso específico de este proyecto, se decidió trabajar con un campo de aplicación del diseño de circuitos integrados, los *OPAMP's*, principalmente para evitar los problemas de coherencia que se presentan cuando se genera texto con diferentes clases de documentos.

Formalmente, esta tarea se basa en la creación de mensajes partiendo de las entradas del sistema o las fuentes de datos; siendo estos los objetos de conocimiento usados por las siguientes etapas de generación de lenguaje. Generalmente, en la construcción de los mensajes se depuran y reunen los datos de entrada y el resultado se expresa en algún lenguaje formal. El presente trabajo no tiene en cuenta esta tarea, porque los mensajes se construyen automáticamente por el sistema a partir de la implementación de algoritmos. Esta labor se deja al humano, quien finalmente se encarga de decidir y aprobar el contenido y la forma del texto.

Planeación del discurso: la planeación del discurso asigna orden y estructura a los mensajes que se transmitirán. Un texto no es un conjunto de elementos de información ordenados

[1]Lenguaje para el formateado de textos (TeX). Fue creado por Donald Knuth y complementado por LaTeX a través de un conjunto de macros escritas en TeX por Leslie Lamport.

aleatoriamente y usualmente existe un esquema para su presentación. En el sistema planteado para este trabajo se facilita al usuario la especificación de esquemas de párrafos, indicando el número de oraciones y la descripción tanto del párrafo como de las oraciones.

Las pruebas al sistema se hicieron usando un párrafo de introducción para publicaciones técnicas relacionadas con el diseño de circuitos integrados. El esquema del párrafo de introducción consta de seis oraciones, en donde la primera indica la idea principal del artículo, la segunda es un refuerzo de la idea principal y las otras cuatro restantes hacen la presentación de las secciones del artículo.

Una buena estructuración facilita la lectura, e incluso la forma y el estilo de escribir texto en una persona. Esta aplicación pretende facilitar la tarea de edición de cada artículo relacionado con el diseño de los circuitos del grupo CIDIC de la Universidad Industrial de Santander(UIS). Indirectamente se espera ayudar a sus miembros a mejorar la forma de editar sus artículos y su dominio del inglés.

Agregación de oraciones: agrupa mensajes en oraciones con el fin de organizar lo propuesto en el plan del discurso. La agregación no siempre es necesaria pues cada mensaje es expresado en una oración por separado, pero en muchos casos, una buena agregación mejora la fluidez y legibilidad de un texto. El sistema no hace esa tarea porque en el plan de discurso, se indican las oraciones que lo componen, por tanto esta labor se realiza de manera manual. Sin embargo, el sistema permite al usuario interactuar con los planes de discurso modificando el orden y el número de oraciones de cada párrafo.

Lexicalización: consiste en seleccionar las palabras y frases que expresarán las ideas del texto. En muchos casos, la lexicalización se hace asignando palabras o frases específicas a un contexto, lo cual implica la existencia de mecanismos de programación en el sistema. El sistema selecciona las frases generadas de los párrafos de introducción por medio del algoritmo de similitud planteado en el trabajo. Las palabras de dichas oraciones seleccionadas se extraen de manera aleatoria y sin tener en cuenta técnicas de inteligencia artificial, lo cuál está lejos del alcance de este trabajo. Tanto las frases como las oraciones son extraídas del *corpus* de texto.

En algunos casos, la fluidez del texto se puede mejorar permitiendo al sistema de GLN variar las palabras usadas en las expresiones, para obtener varias posibilidades de generación. El sistema lo hace por medio de los sinónimos almacenados en el *corpus*, permitiendo a la frase generada tener poca posibilidad de repetirse.

Generación de expresiones de referencia: selecciona las palabras o frases que distinguen cada pieza de información dentro de un texto (párrafos, frases, tablas, imágenes, etc). Está muy relacionada con la lexicalización, siendo además importante en la producción de formas de palabras y frases. Sin embargo, a diferencia de la tarea anteriormente mencionada, ésta es más de identificación, en donde los sistemas necesitan comunicar suficiente información para distinguir una palabra o frase de otra y generalmente se tienen en cuenta aspectos del contexto.

Se puede evidenciar esta tarea claramente en el uso del etiquetador *PoS*. Dicho etiquetador se encarga de asignar una notación a cada una de las palabras teniendo en cuenta su función gramatical, pero esta asignación no siempre es la correcta, por lo que deben ser revisadas y aprobadas por el usuario. Las notaciones resultantes representan la forma gramatical de las oraciones, las cuales se emplean para darle forma a las palabras y ubicarlas dentro en un sentido.

Realización lingüística: aplica las reglas de gramática necesarias para producir el texto, el cual es correcto en sintaxis, forma y ortografía. En este proyecto, el componente sintáctico toma la notación de las formas gramaticales para cada oración (representadas en secuencia

Código HTML:

```
<html><head><title>Sample sentence</title></head>
<body><h1>Sample sentence:</h1>
<p><font color=blue>There are </font><b>20</b> trains each day
<br><font size=1>from:</font> <i>Aberdeen</i>
<br><font size=1>to: </font> <u>Glasgow</u>
<br><h3>That's all!</h3>
</body></html>
```

Resultados de interpretación de etiquetas HTML:

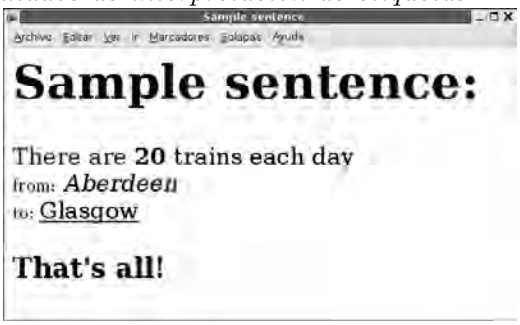


Figura 2.1: *Un ejemplo de HTML.*

de etiquetas *PoS*). El componente morfológico cuida la correcta conjugación de los verbos, porque los recursos lingüísticos empleados *(WordNet y Moby-Thesaurus)* no poseen esta característica. Finalmente, el componente ortográfico coloca con mayúscula inicial la primera palabra de la oración y agrega un punto al final.

Realización estructural: es un proceso de caracter opcional. Consiste en el marcado de las unidades de contenido del texto, tales como palabras, caracteres especiales, párrafos y secciones. Un ejemplo ilustrativo de este proceso es el uso de lenguajes de marcado empleados por algunos sistemas para presentar el texto resultante (HTML, LaTex, etc). El texto, junto con las etiquetas de marcado, se pasan por un intérprete de forma, el cual despliega las funciones de dichas etiquetas (Ver figura 2.1) como: atributos de los caracteres, seccionado del texto y construcciones especiales, entre otros. Se tiene presente en la salida del sistema alguna marcas de *HTML*.

2.2. Módulos

Existen varias formas de construir un sistema que ejecute las tareas de GLN, descritas en la sección anterior. La aproximación más simple es elaborar un módulo por separado para cada tarea y conectar estos módulos secuencialmente. En tal disposición, el módulo de determinación de contenido decide sobre todos los mensajes a incluir; los organiza en un esquema de texto, y sigue operando secuencialmente a través de los siguientes módulos. Las tipologías más comunes en los sistemas de GLN se organizan con los siguientes módulos: planeación de texto, planeación de oraciones y realización de forma (figura 2.2) [42].

Planeación de texto: este módulo contiene las tareas de determinación de contenido y planeación de discurso, descritos anteriormente[2]. En términos operativos, la planeación de texto toma la entrada al sistema de GLN y produce representaciones semánticas, es decir, el

[2]Macroplaneación, planeación de contenido y planeación de diálogo (para sistemas de diálogo).

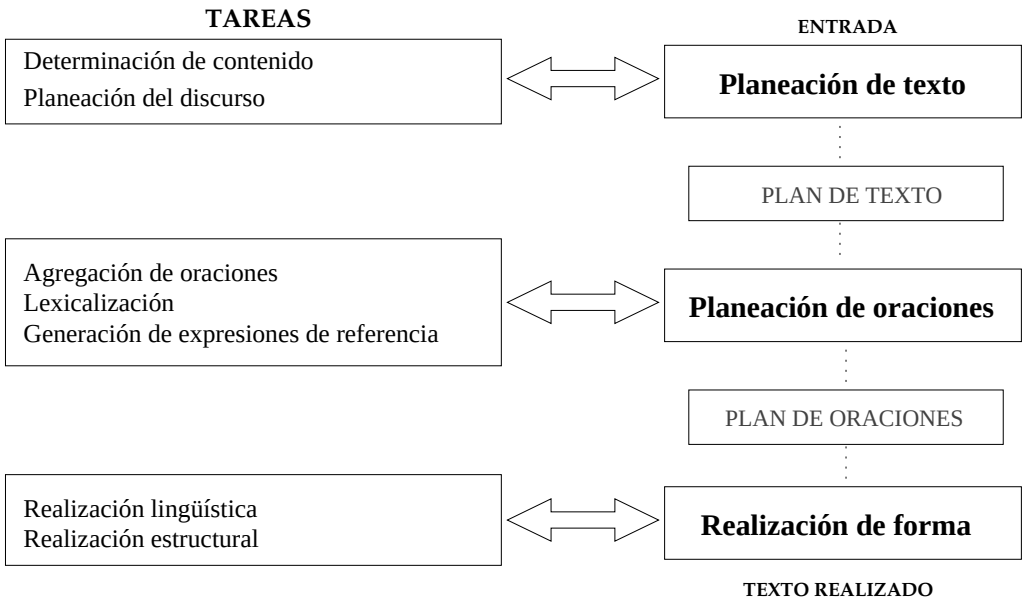


Figura 2.2: *Módulos de un sistema de GLN.*

significado del contenido en términos de esquemas de texto. Además, determina cuál información debe ser comunicada, la organiza en una forma retóricamente coherente y produce un plan de texto.

El sistema origina los planes de texto según los esquemas de párrafo especificados por el usuario y no lo hace de manera automática. Estos esquemas sólamente indican el número y el tipo de las oraciones en cada párrafo. No se tuvieron en cuenta las relaciones retóricas entre las oraciones porque el trabajo se enfocó hacia la generación de oraciones simples. Sin embargo, en las pruebas del sistema cada oración tiene un significado implícito dentro del párrafo (la primera oración es la idea principal, la segunda la idea secundaria y las demás son la presentación de cada sección del artículo).

Planeación de oraciones: la agregación de oraciones, la lexicalización y la generación de expresiones de referencia se combinan en este módulo[3]. Sin embargo, esta combinación no es universalmente aceptada en el campo de la GLN. Por ejemplo Matthiessen, afirma que la lexicalización debe estar integrada con la realización lingüística [44]. A pesar de ello, muchos sistemas aplicados de GLN han elegido unir estas tres tareas dentro de un módulo.

El módulo toma el plan de texto y produce sus especificaciones lingüísticas. Determina cómo la información debe ser transmitida y compara las relaciones retóricas e ideas del texto con el contenido de las palabras y sus conexiones gramaticales formando así el plan de oración.

La herramienta del presente trabajo especifica la manera de expresar las oraciones mediante el proceso realizado por el algoritmo de similaridad de frases, pero como ya se mencionó, no se tienen en cuenta las relaciones retóricas entre las oraciones porque está más allá del alcance de este trabajo. El sistema arroja el plan de oración en una secuencia de etiquetas PoS, las cuales tienen asociados sinónimos y un indicador de similaridad con base en el cuál se selecciona la frase a generar.

[3]También conocido como microplaneación.

Realización de forma: esta tarea involucra procesamiento sintáctico, morfológico y ortográfico, así como la generación de las palabras teniendo en cuenta la estructura del texto[4]. Además, expresa el contenido de las palabras y las relaciones gramáticas según el lenguaje seleccionado. Genera el texto final y, si se especifica, sus respectivas marcas de estructura.

El procesamiento de sintáxis, forma y ortografía se hace con las palabras y formas de las oraciones aprobadas por el humano cuando está creando el corpora de texto, así el sistema ya toma en cuenta estas características preocupándose por la generación de palabras. Sin embargo, se hizo la implementación de un conjugador de verbos dada la necesidad de refinar las oraciones y en vista de la ausencia de herramientas de código abierto que pudieran ayudar en este sentido. El contenido y las relaciones gramaticales están en inglés y al final se presenta el texto con etiquetas HTML.

Representaciones intermedias: considerando la arquitectura global de un sistema de GLN, existe un problema importante en el diseño y consiste en saber cuál es la manera de representar las diferentes entradas y salidas entre los módulos del sistema. La entrada inicial del sistema es dependiente de la aplicación y la salida final es el texto (quizás con algun marcado lógico o físico para propósitos de presentación, tales como las etiquetas HTML). El desarrollador en GLN debe especificar la representación interna a transferirse desde el módulo de planeación de oraciones hasta el módulo de realización de forma. La salida del planeador del texto se denomina plan de texto y la salida del planeador de oraciones para cada oración es llamado plan de oración.

Planes de texto: el plan de texto se construye extrayendo datos desde una fuente de conocimiento. Se agrupa la información por medio de relaciones retóricas y se restringe el alcance de la planeación de oraciones. De ser posible, también se parametriza la ubicación y el tamaño de palabras y frases. El plan de texto del sistema se forma a partir de las especificaciones dadas en los esquemas de párrafo previamente indicados por el usuario. El sistema no hace especificación sobre las oraciones en su forma y número de palabras, pero si en la posición de cada oración.

Planes de oraciones: una amplia variedad de mecanismos y notaciones para planes de oraciones ha sido propuesta y usada en la literatura, de las cuales las más comunes son probablemente las plantillas denominadas representaciones abstractas. Los sistemas de plantilla representan las oraciones como estructuras de texto y parámetros que necesitan ser insertados dentro de ellas. Los sistemas clásicos de plantilla simplemente insertan el parámetro dentro de la estructura de texto sin hacer ningún procesamiento posterior.

Otra forma usual de representar los planes de oraciones es usar un lenguaje de representación abstracta para especificar las palabras contenidas (nombres, verbos, adverbios y adjetivos) en una oración, y cómo se relacionan. Uno de los más populares lenguajes de representación abstracta para planes de oraciones es el *SPL* [45]. En nuestro caso el lenguaje de representación abstracto son las etiquetas PoS del Penn TreeBank, asignada a cada palabra en las oraciones. Aparte de estas etiquetas, se especificaron otras propias, por la necesidad de hacer notaciones sobre los términos fijos, palabras técnicas y expresiones agrupadas. Toda la información de los planes de oración está organizada en una base de datos dónde se pueden relacionar las etiquetas Pos y los sinónimos.

[4]Llamada también generación de forma.

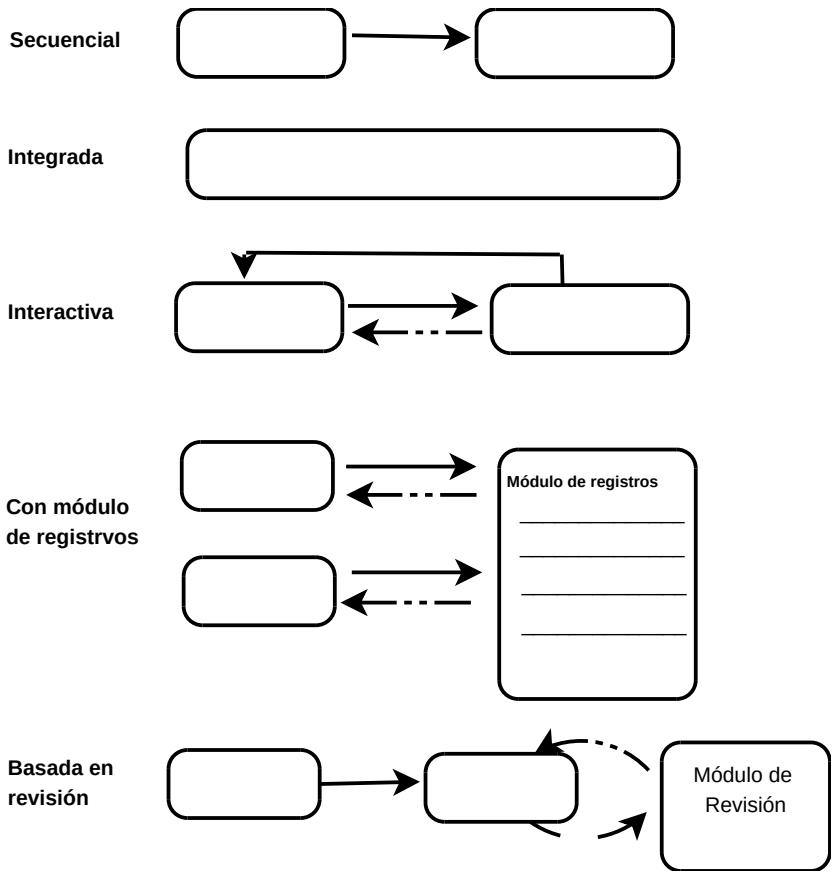


Figura 2.3: *Arquitecturas de sistemas de GLN.*

2.3. Diferentes tipologías en sistemas GLN

Aunque muchos sistemas de generación de texto emplean las mismas subtareas y módulos, ellos difieren en la forma como se organizan estos módulos. Tal disposición define la arquitectura usada en todo el sistema. La figura 2.3 presenta un esquema de las diferentes arquitecturas utilizadas en sistemas de GLN. Dichas arquitecturas son: secuencial, integrada, interactiva, con módulo de registros y basada en revisión.

Arquitectura secuencial: la arquitectura secuencial incorpora un flujo de información en un solo sentido a través de todos los módulos del sistema. En una arquitectura estrictamente secuencial no hay realimentación entre los módulos. Los sistemas MUMBLE [29] y TEXT [28] son ejemplos representativos de arquitectura secuencial.

Una arquitectura secuencial tiene como ventaja la simplicidad. Cada módulo está encapsulado al máximo porque recibe las entradas desde una fuente y envía la salida a un destino. Las representaciones entre cada nivel del sistema son asumidas como correctas y completas hasta llegar a la expresión generada. Sin embargo, dada su rigidez, esta arquitectura puede generar problemas, en especial cuando es posible la interacción mutua entre varios niveles lingüísticos, siempre y cuando ésta posiblemente se omita.

Arquitectura integrada: KAMP es el mejor ejemplo conocido de una arquitectura integrada, quizás la aproximación más radical cuando no existen modulos en el sistema [21]. Todas las decisiones son tomadas en una sola parte, haciendo que el procesamiento sea estructurado como una caja negra. De esta forma, la generación tiene lugar en un desarrollo

continuo manejado por objetivos y basado en restricciones. Particularmente, las técnicas para satisfacer las restricciones pueden variar, por ejemplo con planeación, programación lógica o métodos de búsqueda.

Una arquitectura integrada permite representar y procesar diferentes tipos de conocimiento de manera uniforme, lo cual se considera un beneficio debido a la organización a la hora de construir de fuentes de conocimiento. No obstante, ésto también puede ser desventaja si las distintas fuentes de conocimiento tienen interferencias difícilmente detectables. Lo anterior dificulta el mantenimiento, la transparencia y la construcción de un sistema eficiente a gran escala.

Arquitectura interactiva: las arquitecturas interactivas involucran realimentación entre las diferentes fases del procesamiento en ciertos puntos del sistema. Por consiguiente, entre los módulos aparecerá la representación intermedia generada parcialmente que puede devolverse al módulo anterior como realimentación. Un ejemplo, es el sistema PAULINE [22].

Existen además varias maneras de implementar la función de realimentación entre los módulos. POPEL [16] emplea un manejador de solicitudes para proporcionar la interfase necesaria entre los módulos; en el sistema IGEN [17], la realimentación se realiza a través de anotaciones indicando la cantidad de contenido a generar según la elección de palabras específicas. Con estas anotaciones, el módulo encargado de realizar el texto determina cuales funciones satisfacen las especificaciones secundarias.

Arquitectura con módulo de registros: otra forma de interacción es presentada en las arquitecturas con módulo de registros para la generación, como en el sistema DIOGENES [18]. En tal arquitectura, los módulos proporcionan información sin depender necesariamente del conocimiento usado por los otros módulos. Cada módulo revisa los registros buscando la información necesaria y escribe su propia salida.

Una ventaja de esta arquitectura es la identificación de un conjunto de fuentes de conocimiento, las cuales dejan información útil para beneficiar el mantenimiento del sistema y proporcionar evidencia sobre su rango de cobertura. Adicionalmente, los sistemas con módulo de registros permiten procesamiento flexible: la ejecución temporal de varios módulos se originar en diferente orden, siempre y cuando en el módulo de registro exista la información necesaria para poder ejecutarse. Sin embargo, la interacción entre las fuentes de conocimiento y el módulo de registro necesita ser soportada por algún mecanismo para incrementar la eficiencia y resolver conflictos. En general, es necesario controlar el conocimiento extra para asignar prioridades a las fuentes de conocimiento.

Arquitectura basadas en revisión: las arquitecturas basadas en revisión presuponen una forma limitada de realimentación, mediante módulos de monitoreo los cuales inspeccionan las estructuras intermedias. Así, el texto originado en la secuencia puede modificarse para reparar cualquier deficiencia detectada. En el sistema KDS estos módulos son correctores [19], mientras en WEIVER [20] las decisiones iniciales que conducen a resultados erróneos se deshechan.

2.4. Nuestra arquitectura

Dado el carácter de aproximación, este trabajo se acerca más a la arquitectura secuencial porque se pueden identificar módulos que cumplen una función y no existe realimentación entre ellos. La figura 2.4 presenta el diagrama de la tipología de nuestro sistema. El módulo de planeación de texto toma como entrada las palabras solicitadas por el sistema incluidas en el texto y arroja como resultado un esquema de párrafo. Este esquema es tomado por el módulo

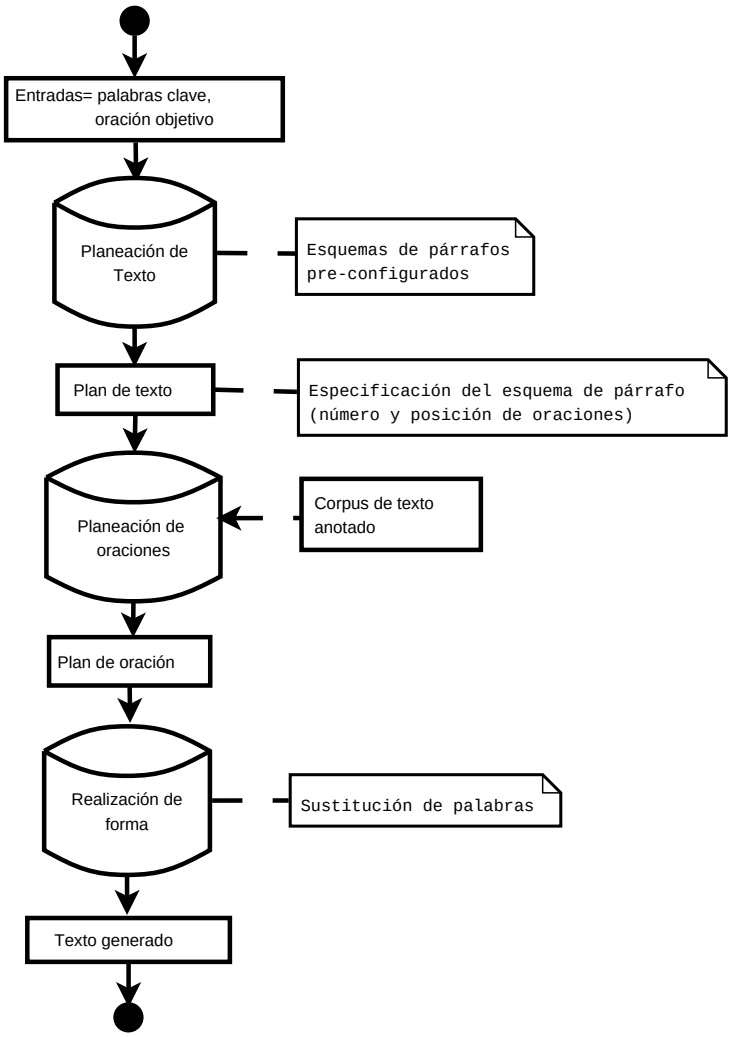


Figura 2.4: *Arquitectura de nuestro sistema.*

de planeación de oraciones para generar las secuencias de etiquetas PoS de cada oración, con cual se indica al generador de forma las palabras a generar. Al no existir una realimentación entre los módulos del sistema, se impide ubicar el trabajo en las arquitecturas interactiva y basada en revisión. Los módulos dependen de la información transmitida entre ellos, por tanto una arquitectura con módulo de registro no es aplicable. Finalmente es muy difícil concebir el sistema en términos de un sólo módulo, descartándose el uso de una arquitectura integrada.

2.5. Consideraciones del capítulo

En este capítulo se explicaron los componentes básicos de un sistema de generación de texto, entradas, salida, tareas, modulos y tipologías. Las entradas constan de alguna representación de información como tablas, informes, datos estadísticos de un objetivo comunicativo, junto con varias fuentes de conocimiento. La entrada se representa como el conjunto **c,o,u,d**, dónde **c** es la fuente de conocimiento empleada, **o** es el objetivo comunicativo, **u** es el modelo del usuario y **d** es el modelo del discurso. La salida es básicamente texto en lenguaje natural, es decir, un producto en lenguaje natural comprendido por un ser humano. Las tareas de un

sistema de GLN son caracterizadas como un proceso que vincula algunos datos de entrada a una salida de texto. Las siete tareas básicas son las siguientes: determinación de contenido, planeación del discurso, agregación de oraciones, lexicalización, generación de expresiones de referencia, realización lingüística y realización estructural. Los módulos son los métodos creados para construir un sistema de GLN basado en sus tareas. Los módulos más usados son planeación de texto, planeación de oración y realización de forma, existen además otros modelos para las representaciones intermedias entre dichos módulos. Las tipologías representan la forma en como los módulos de un sistema de generación de texto se encuentran organizados. Se clasifican en secuencial, integrada, interactiva, basada en revisión y módulo de registros.

Este marco de estudio se comparó con el trabajo realizado para dar claridad a los conceptos, sentando así las bases teóricas que contribuyan a la comprensión de la GLN. Sin embargo, dada la extensión de este campo del conocimiento, no se hizo una revisión exhaustiva y detallada sino general, tratando de ser lo más claro y conciso posible. Hay aspectos de los componentes de los sistemas de GLN no contemplados en el alcance de este capítulo, no obstante, quedan los fundamentos para motivar la profundización en estos temas y la realización de futuros trabajo relacionados. En el siguiente capítulo se hace una revisión sobre algunas herramientas de uso libre en computación lingüística que fueron empleadas para realizar un primer prototipo de generación de texto.

Capítulo 3

Herramientas de generación, una revisión

Las consideraciones estimadas en este capítulo, se relacionan con los aspectos teóricos de los componentes relacionados con este trabajo. Inicialmente, se hace una revisión bibliográfica, de trabajos con características similares a las planteadas en este documento, a partir de ello, se encuentran desarrollos asociadas a las tareas de GLN, como la teoría de la estructura retórica, los recursos lingüísticos y el problema de la ambigüedad semántica. Con lo anterior, se plantea un prototipo inicial de software, cuyos resultados evidencian los problemas y las limitaciones de producir textos automáticamente.

3.1. Generadores de texto existentes

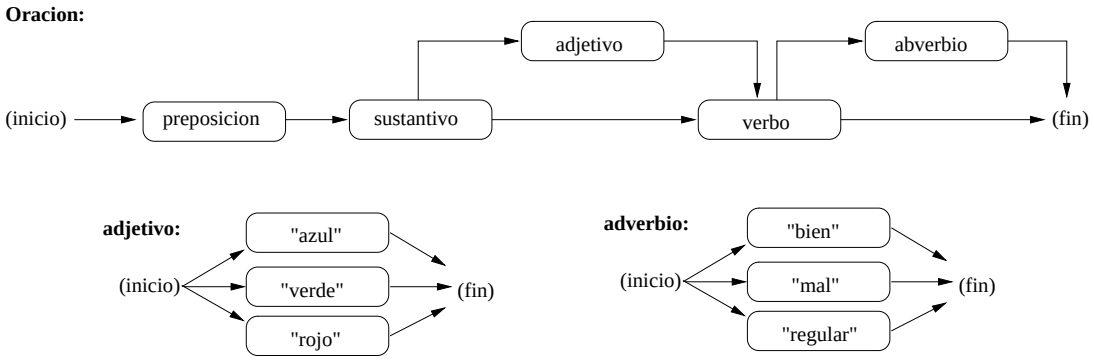
Los generadores de texto modernos se fundamentan en diversas técnicas y son construidos para un lenguaje y dominio específico de conocimiento. Al hacer la revisión en la literatura, las herramientas más importantes encontradas para la edición y producción de texto, especializadas en artículos para publicaciones técnicas en inglés son el *Dada Engine* [35] y *SciGEN* [40]. *Dada Engine* es un sistema para crear texto aleatorio a partir reglas gramaticales. Las reglas están especificadas en un lenguage conocido como *pb*, con una sintáxis similar al formato *Backus-Naur (BNF)*.[1] El sistema está soportado por el principio de gramáticas recursivas, las cuales corresponden a diagramas empleados para representar la forma de elaborar una oración usando trayectorias y operaciones. Las redes de transición recursivas son representadas textualmente como reglas, ver ejemplo en la figura 3.1. El texto resultante es intérpretado por la herramienta, que evalúa las reglas y emite la salida.

Por otro lado, *SciGEN* produce randómicamente un artículo de investigación relacionado con ciencias de la computación, utilizando gramáticas independientes del contexto. El documento construído incluye gráficas, figuras, tablas y citas bibliográficas. Su fin no está enfocado a la coherencia del texto[2], pero el valor de este trabajo radica en el estudio hecho sobre los patrones de escritura comunes en una gran cantidad de artículos técnicos en inglés, a partir de los cuales fueron formuladas las gramáticas.

La figura 3.2 presenta un ejemplo del uso de *SciGEN* para elaborar textos. Inicialmente,

[1] Metasintáxis usada para expresar gramáticas libres de contexto (conjuntos de palabras de longitud finita a partir de un alfabeto finito) [57].

[2] Su intención es meramente lúdico. Los artículos eran enviados a conferencias para comprobar si podían ser aceptados, evidenciando así fallas en la recepción de estos trabajos.




```
Oracion: preposicion [ adjetivo | "" ] sustantivo [ adverbio | "" ] ;
adjetivo: "azul" | "verde" | "rojo" ;
adverbio: "bien" | "mal" | "regular" ;
```

Figura 3.1: *Diagrama de una gramática recursiva.*

la herramienta selecciona al azar un esquema de artículo entre varios predeterminados, dentro del cual está especificado las secciones que contendrá (*SCIPAPER*); cada sección es asociada a un formato particular usado como plantilla (por ejemplo, *SCI_ABSTRACT*) para formar el esquema de párrafo, y del mismo modo cada elemento de la plantilla, contiene palabras finales u otras plantillas, que desempeñan el papel de gramáticas, las cuales pueden contener palabras y símbolos terminales.

En un principio se plantea la idea de utilizar *SciGEN* en la implementación de un generador aleatorio de artículos dedicado al tema del diseño de circuitos integrados, pero ésta idea es rechazada muy pronto al notar la falta de algún elemento ''inteligente"que otorgue soporte durante el proceso de generación.

3.2. Teoría de la estructura retórica

La Teoría de la Estructura Retórica (*RST*[3]) ofrece una interpretación de la coherencia textual independiente de las formas léxicas y gramaticales [58]. Coherencia se define como la ausencia de secuencias ilógicas y de lagunas, en otras palabras, cada parte tiene una razón aceptable para su existencia evidente a los lectores, y además hace parecer que el contenido está completo.

La *RST* enfatiza en la función de cada segmento textual y establece un análisis para cualquier documento, explicando los contenidos expresados por el autor. Para realizarlo, se utilizan cuatro objetos: las relaciones entre dos tipos de texto, los esquemas para establecer la manera de analizar y relacionar los fragmentos de información, la aplicación de los esquemas y la estructura del texto [59].

A partir de las relaciones presentadas en la figura 3.3, es posible elaborar el análisis de un texto extraído de un artículo de la revista *Scientific American* y representado en el árbol retórico. Lo primero es identificar el título y el párrafo divididos en unidades numeradas para el análisis. Para el analista, generalmente denominado observador en los artículos sobre *RST*, las primeras dos unidades (las que explican los términos lactosa y lactasa) facilitan la comprensión del resto del texto. La unidad número 2 proporciona detalle adicional sobre

[3]Del inglés *Rethorical Structure Theory.*

Representación de las reglas:

```
SCIPAPER-> {SCI_ABSTRACT SCI_INTRO SCI_MODEL SCI_IMPL SCI_EVAL SCI_RELWORK SCI_CONCL}
:
SCI_ABSTRACT-> {SCI_INTRO_A SCI_ABSTRACT_A SCI_INTRO_THESIS}
:
SCI_INTRO_A-> Many SCI_PEOPLE would agree that, had it not been
for SCI_GENERIC_NOUN , the SCI_ACT might never have occurred XXX

SCI_ABSTRACT_A->SCI_AB_A_START the SCI_ACT SCI_AB_A_END XXX
:
SCI_AB_A_START-> Given the current status of SCI_BUZZWORD_ADJ SCI_BUZZWORD_NOUN,
                 SCI_PEOPLE SCI_ADJ_ADV desire
:
SCI_ACT-> SCI_ACT_A SCI_THING that SCI_EFFECT
SCI_ACT_A-> synthesis of
SCI_THING-> telephony
SCI_EFFECT-> paved the way
:
SCI_AB_A_END-> for the SCI_ADJ of SCI_FIELD XXX
:
SCI_INTRO_THESIS-> SYSNAME, our new SCI_SYSTEM for SCI_GENERIC_NOUN, is the solution
                   to all of these SCI_PROBLEM_PL XXX
```

Ejemplo de un texto generado para *SCI_ABSTRACT* (Resumen del artículo):

```
Many researchers would agree that, had it not been for context-free grammar,
the improvement of spreadsheets might never have occurred. Given the current
status of mobile communication, researchers urgently desire the synthesis of
telephony that paved the way for the deployment of consistent hashing. GALT,
our new heuristic for RAID, is the solution to all of these problems.
```

Figura 3.2: *Gramáticas del SciGEN.*

el tema presentado en la unidad número 1, también se establece que las unidades 4 y 5 se encuentran en una relación neutra de contraste. Todas estas observaciones se formulan en términos de la intención del autor, por ejemplo, se postula la intención de hacer reconocer al lector las situaciones expresadas en las unidades 4 y 5 como muy similares, pero diferentes en un aspecto señalado por el autor.

Además del análisis de discursos, existen importantes desarrollos en la lingüística computacional que emplean la RST, como el parseo de texto, resumen, creación de recursos lingüísticos y la traducción automática. Su aplicación más común se ha dado en generación de lenguaje natural [13], utilizando las relaciones de discurso como componentes de planeadores de texto y de contenido. En [10] se proporciona un resumen de los primeros trabajos en generación automática de texto, en los cuales una de las aplicaciones descritas es el desarrollo de una interfaz para una base de datos con información de barcos y sus coordenadas, que convierte relaciones retóricas en planes de texto.

Se realiza la revisión bibliográfica, para comprender el tratamiento dado al problema de la coherencia en la producción del texto, lo cual motiva la búsqueda de software relacionado con la *RST*, que puedan adaptarse a los requerimientos para un generador de artículos técnicos. El *ILEX* [60], generador de textos que emplea árboles de estructura retórica como plan de texto, hace parte de las aplicaciones más representativas, sin embargo, no es posible realizar la adaptación de esta herramienta debido a la imposibilidad de obtener su código fuente.

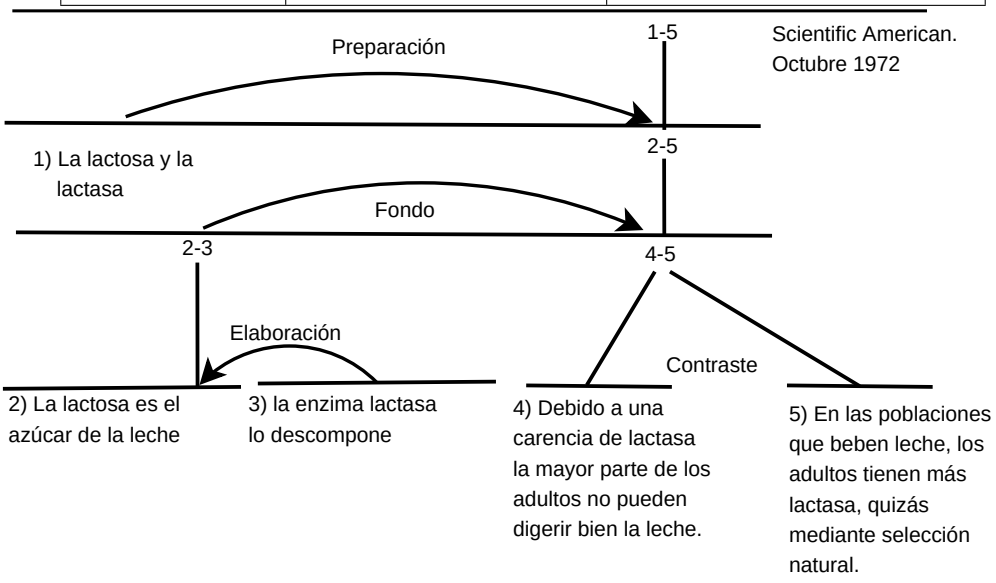
En vista de ésta dificultad, se plantea la construcción de una herramienta para realizar automáticamente el análisis retórico de escritos para artículos técnicos, cuyo resultado pueda emplearse en generación. Además es necesario elaborar un estudio de las formas de escritura

Relaciones núcleo - satélite:

Relación	**Núcleo**	**Satélite**
Elaboración	Información básica	Información adicional
Fondo	Texto del cuál se facilita la compresión	Texto que facilita la compresión
Preparación	Texto que va a ser representado	Texto que prepara al lector para anticipar e interpretar el texto que va a ser presentado

Relaciones multinucleares:

Relación	**Unidad**	**Segunda unidad**
Contraste	Una opción	La otra opción



Título: 1) La lactosa y la lactasa	2) La lactosa es el azúcar de la leche;	3) la enzima lactasa lo descompone.	4) Debido a una carencia de lactasa la mayor parte de los adultos no pueden digerir bien la leche.	5) En las poblaciones que beben leche, los adultos tienen más lactasa, quizás mediante selección natural.	*Scientific American*, Octubre 1972.

Figura 3.3: *Ejemplo de análisis retórico adaptado de [58].*

según el estilo adoptado por autores de documentos de este tipo, con el fin de encontrar patrones cuantificables, numerables y clasificables dentro de categorías retóricas análogas a las planteadas por la *RST*. Luego, puede diseñarse un motor de inferencia capaz de detectar relaciones retóricas a partir de entradas de texto y emplearlas posteriormente en una estrategia de producción de textos, lo anterior limita el trabajo a un estudio retórico, perdiendo así la concepción original de éste.

3.3. Recursos lingüísticos

Actualmente existe gran cantidad de texto en formato digital, desde documentos publicados hasta catálogos bibliográficos, lo cual ha originado la existencia de diversas y novedosas aplicaciones en lingüística computacional, dada la necesidad de disponer de fuentes de conocimiento especializadas para realizar el tratamiento de toda esa información [61]. Entre las herramientas desarrolladas se destacan los *corpus* de texto, los etiquetadores de partes de voz, las bases de datos léxicas y los desambiguadores semánticos.

3.3.1. *Corpus* de texto

Un *corpus* lingüístico es una colección de textos representativos, producidos a partir del estudio de un idioma, dialecto o subconjunto de un lenguaje utilizados para el análisis lingüístico [62]. Los primeros *corpus* fueron creados en los años 60 y han alcanzado su mayor popularidad recientemente, debido principalmente al éxito de métodos estadísticos, como, los n-gramas, el modelo del espacio vectorial, al incremento relacionado con el cálculo y la capacidad de almacenamiento de los sistemas computacionales.

Dependiendo del criterio considerado, pueden realizarse varias clasificaciones; respecto al material contenido se pueden agrupar en dos grandes tipos, textuales y orales. Los *corpus* de textos orales consisten en transcripciones ortográficas de grabaciones realizadas a diversas situaciones en la lengua hablada. Éstas pueden codificarse de acuerdo a un formato estándar, por ejemplo el *Text Encoding Initiative* (TEI) presentando distintos niveles de anotación. Por su parte, los *corpus* textuales son simplemente reproducciónes de escritos.

También puede establecerse una distinción de los *corpus* de acuerdo al objetivo, en generales y específicos. Un *corpus* de fines generales constituye una fuente de información textual para un idioma en diversas aplicaciones. Los *corpus* con fines específicos son creados en respuesta a un propósito particular, como el estudio de aspectos concretos de la gramática o del léxico de un idioma, la extracción de datos estadísticos, el estudio del comportamiento lingüístico de una determinada población de hablantes o el desarrollo y evaluación de sistemas de procesamiento del lenguaje natural. Un *corpus* creado con fines específicos puede reutilizarse para tareas distintas de las previstas, siempre y cuando sus características y diseño se adecúen a ellas.

Finalmente, el *corpus* puede estar lingüísticamente anotado o no anotado. Un *corpus* no anotado sólo dispone de la colección de textos sin ninguna información adicional, siendo la tipología más frecuente, por ejemplo el *Brown* corpus [50]. Los *corpus* codificados o anotados están formados por textos con determinada información agregada de forma manual o automática, acerca de las palabras y frases que contiene, éstas pueden señalar la estructura de los textos por medio de etiquetas especiales para indicar el título, los capítulos, y otros elementos del texto (codificación) o aspectos puramente lingüísticos, como la categoría gramatical, la estructura sintáctica, entre otros (anotación). El proceso de anotación de *corpus*

Tabla 3.1: *Conjunto de etiquetas de partes de voz para el Penn TreeBank.*

1.	CC	Conjunción coordinante	25.	TO	*to*
2.	CD	Número cardinal	26.	UH	Interjeción
3.	DT	Determinante	27.	VB	Verbo en forma base
4.	EX	*there* existencial	28.	VBD	Verbo en pasado
5.	FW	Parabra foránea	29.	VBG	Verbo en gerundio o participio presente
6.	IN	Preposición	30.	VBN	Verbo en participio pasado
7.	JJ	Adjetivo	31.	VBP	Verbo en presente sin tercera persona
8.	JJR	Adjetivo comparativo	32.	VBZ	Verbo en presente con tercera persona
9.	JJS	Adjetivo superlativo	33.	WDT	Determinante *wh*
10.	LS	Item de lista	34.	WP	Pronombre *wh*
11.	MD	Modal	35.	WP$	Pronombre posesivo *wh*
12.	NN	Sustantivo singular	36.	WRB	Adverbio *wh*
13.	NNS	Sustantivo plural	37.	#	Numeral
14.	NNP	Nombre propio singular	38.	$	Signo de dólar
15.	NNPS	Nombre propio plural	39.	.	Punto final de oración
16.	PDT	Predeterminante	40.	,	Coma
17.	POS	Terminación posesiva	41.	:	Dos puntos
18.	PRP	Pronombre personal	42.	(	Paréntesis izquierdo
19.	PP$	Pronombre posesivo	43.	)	Paréntesis derecho
20.	RB	Adverbio	44.	”	Comillas dobles
21.	RBR	Adverbio comparativo	45.	’	Comilla simple izquierda
22.	RBS	Adverbio superlativo	46.	”	Comillas dobles izquierda
23.	RP	Partícula	47.	’	Comilla simple derecha
24.	SYM	Símbolo científico o matemático	48.	”	Comillas dobles derecha

se puede realizar con el etiquetado por partes de voz [4], el cual adiciona información relativa a la categoría propia de cada una de las palabras en un texto, por ejemplo, verbo, sustantivo, adjetivo, entre otras.

Un ejemplo representativo de *corpus* anotado sintácticamente es el *Penn TreeBank* [66], el cual consta de aproximadamente 4,5 millones de palabras de inglés norteamericano, anotado con información de partes de voz y con un esquema de representación sintáctica. Producto del etiquetado del *corpus*, se establecen determinadas etiquetas de partes de voz, a partir de una simplificación de las usadas en otros *corpus*. El conjunto de partes de voz del *Penn TreeBank* se muestra en la tabla 3.1. El proceso inicialmente asigna la etiqueta de parte de voz automáticamente y luego se corrige por una persona.

Un trabajo derivado del *PTB* es el *Penn Discourse Penn TreeBank, PDTB,* que contiene anotaciones de conectivos de discurso entre argumentos, basados en un millón de palabras del PTB [67]. Este recurso profundiza el estudio de las relaciones de coherencia, tratando los conectivos de discurso como predicados entre dos objetos abstractos,por ejemplo, un evento, un estado, o ona proposición. Los conectivos están clasificados en explícitos, términos claramente definidos, e implícitos, donde pueden deducirse relaciones entre dos argumentos sin estar indicadas.

[4]También conocidos como categorías léxicas o clases de palabra, las partes de voz son categorías lingüísticas que definen el comportamiento sintáctico y morfológico de cada palabra.

Durante este trabajo, se ha realizado una revisión de este recurso y las posibles maneras de aplicarlo al proceso de generación. Los resultados obtenidos por los autores de PDTB evidencian mejoras en la coherencia de los textos, dando mayor fluidez a los párrafos mediante la asignación apropiada de los conectivos entre sus oraciones. Sin embargo, para emplear esta herramienta es necesario adquirir el *PTB*, al cuál no se tiene acceso debido a que es de uso privativo, con costos asociados no contemplados en el proyecto. Se han buscado herramientas relacionadas con el *PTB* (el más representativo es el *Wall Street Journal* corpus), con la que se esperan proporcionar automáticamente las etiquetas de parte de voz correspondientes a cada palabra para un texto sin anotaciones, teniendo así un *corpus* anotado con características similares y compatible con el *PDTB*. Sin embargo, es necesario realizar anotaciones sintácticas al texto y las herramientas encontradas no poseen esta función.

La elaboración de un generador automático de textos considerando el dilema de la coherencia puede apoyarse en un *corpus* anotado que contenga información de las relaciones de discurso. Debe realizarse un estudio formal sobre relaciones de discurso, lo cuál requiere de expertos en lingüística y estaría más allá del alcance de este trabajo. Esta persona debe tener el suficiente conocimiento en el área de estudio, para poder reconocer los patrones de escritura de los expertos en la materia por medio de entrevistas y observaciones, luego realizar una representación de sus resultados, con el fin de emplearlos como recurso computacional en generación. Todo este trabajo implica un retraso en el cronograma estimado, además se crea gran dependencia en la calidad de la información suministrada por el analista. En tal caso, se sugiere elaborar primero el estudio y tener los resultados al alcance para poder iniciar un proyecto de generación de textos con estas características.

La idea inicial de generar texto aleatoriamente para dar solución al problema de coherencia textual con *RST*, *corpus* anotados y relaciones de discurso, se abandona dada las dificultades mencionadas, sin embargo, sirven de fundamento conceptual para encaminar la revisión en el área hacia un nuevo enfoque.

3.3.2. Etiquetadores de partes de voz

El etiquetado gramatical[5] asigna marcas a cada palabra de un texto con la parte de voz correspondiente, respecto a la definición de la palabra o al contexto [68]. Etiquetar no es sólo poseer una lista de palabras con su parte de voz, ya que pueden tener más de un significado, lo cual es usual en los lenguajes naturales,[6] donde gran parte de las palabras son ambiguas.

Dependiendo del método empleado, pueden clasificarse en estocásticos, basados en reglas y con motor de inferencia [69]. Los etiquetadores estocásticos usan modelos de *Markov*, árboles de decisión, modelos de n-gramas o modelos de máxima entropía. Los etiquetadores basados en reglas emplean conocimiento lingüístico en forma de restricciones para establecer las combinaciones de etiquetas aceptables, y los etiquetadores basados en motores de inferencia, toman un *corpus* anotado y realizan el entrenamiento de un mecanismo de inferencia, sea un sistema experto, red neuronal, entre otros, capacitándolos para realizar anotación sobre *corpus* no anotados. En este sentido, se recurre al uso del *SVMTool* [70] como herramienta de código abierto para generación del sistema planteado.

El *SVMTool* es un generador secuencial de etiquetas basado en máquinas de soporte vectorial (*SVM*[7]), esta herramienta posee tres componentes principales, el modelo de aprendizaje

[5] También llamado en inglés *Part-of-speech tagging ,POS tagging or POST*

[6] Opuesto a los lenguajes artificiales donde cada palabra es única y representa un significado preciso, por ejemplo los lenguajes de programación.

[7] Del inglés *Support Vector Machines*.

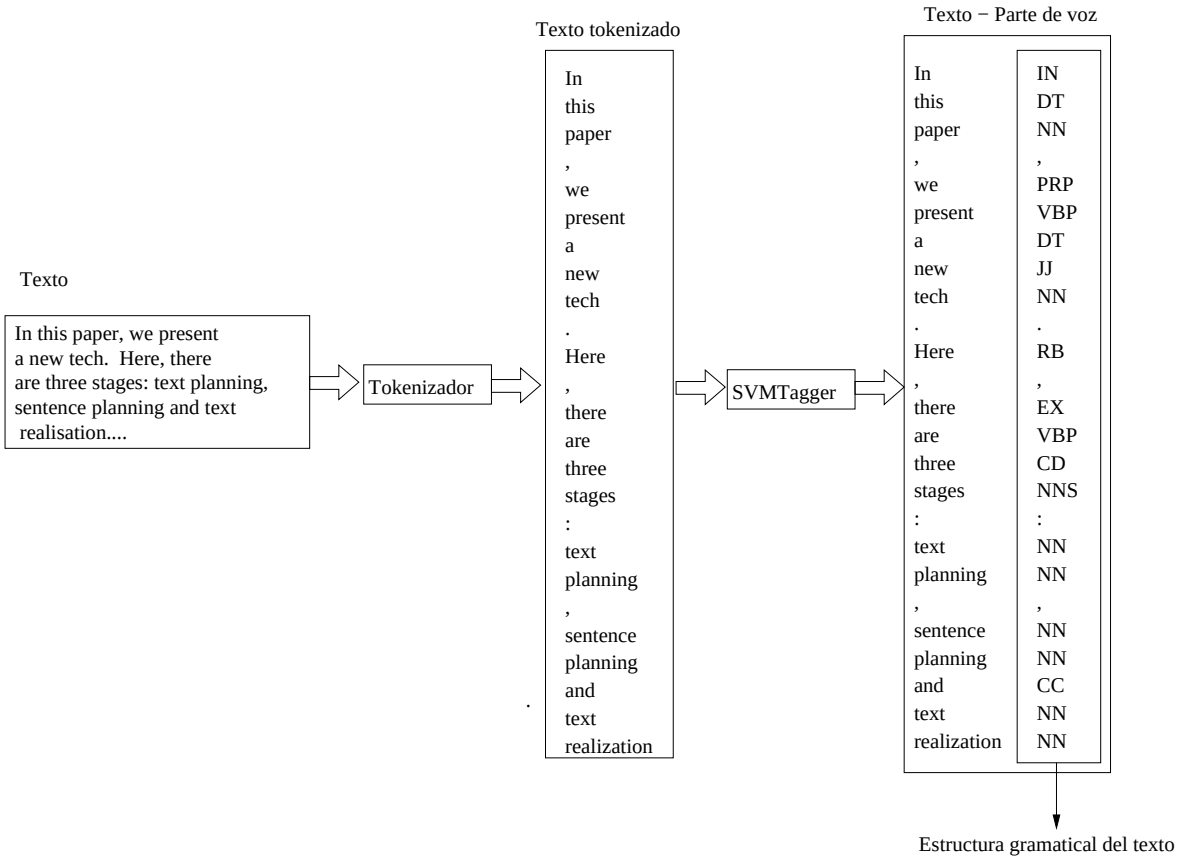


Figura 3.4: *Empleo del SVMTagger.*

SVMTlearn, el etiquetador *SVMTagger* y el evaluador *SVMTeval*. El modelo de aprendizaje está formado por un conjunto entrenado de textos anotados y no anotados generados por la herramienta computacional *SVM-light* [71], clasificándolos en *SVM*. El *SVMTagger* trabaja con las palabras y símbolos de un texto separados por líneas y produce la etiqueta de parte de voz correspondiente, con base en el modelo construido por el *SVMTlearn*. Finalmente el *SVMTeval* mide los resultados obtenidos contra los datos de los *corpus* utilizados.

La forma de operación y la salida del *SVMTagger* está representado en el esquema de la figura 3.4. El texto no anotado debe estar separado por líneas respecto a cada palabra y símbolo. El resultado consta del texto junto con la correspondiente parte de voz asociada automáticamente por el etiquetador. El modelo de aprendizaje empleado por el etiquetador se ha generado a partir del texto anotado del *Penn TreeBank*, lo cual indica que las etiquetas de partes de voz asociadas a los textos corresponden con los de este *corpus*.

Las etiquetas de partes de voz formadas por el *SVMTagger* a partir de un texto, presentan información útil para ser empleada en un proceso posterior de producción de nuevos textos, ya que reflejan estructuras gramaticales y bajo este principio, es posible plantear una estrategia básica de generación de textos. Los textos de artículos técnicos ya editados se pasan al *SVM-Tagger* para obtener su secuencia de partes de voz, por medio de la cual es posible realizar texto nuevo con características sintácticas y morfológicas similares por sustitución simple de cada palabra por sinónimos. Se requiere entonces, de fuentes lingüísticas que proporcionen los sinónimos para cada término.

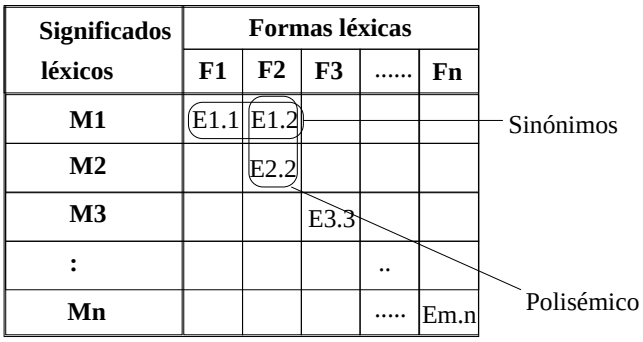


Figura 3.5: *Empleo del Matriz de vocabulario de WordNet*

3.3.3. Bases de datos léxicas

WordNet es un recurso lingüístico de uso libre para el idioma inglés organizado con base en significados en forma de *thesaurus*[8] [73]. Posee las propiedades de un diccionario convencional como descripciones semánticas, deletreado, pronunciación, formas derivadas, etimología, información gramatical, usos, sinónimos y antónimos, además de proveer descripciones basadas en conceptos y relaciones psicolingüísticas entre palabras.

WordNet divide el lexicón[9] en cinco categorías: nombres, verbos, adjetivos, adverbios y partículas; ésta organización hace redundante la información, debido principalmente a los casos donde una palabra pertenece a más de una categoría. El fundamento teórico del sistema tiene su origen en el principio de la matriz de vocabulario, donde las columnas contienen todas las palabras (formas léxicas) de un idioma, mientras que las filas poseen todos los significados. Una entrada de una celda de la matriz implica usar la forma léxica de una columna en el contexto apropiado para expresar el significado de esa fila. En el esquema representado por la figura 3.5, la entrada E1.1 implica usar la forma léxica F1 para expresar el significado M1. En caso de existir dos entradas en la misma columna, la forma léxica es polisémica; si hay dos entradas en la misma fila, las dos formas léxicas son sinónimas.

La representación de los conceptos está basada en una entidad denominada *synonym set*. Un *synonym set*, abreviado *synset*, es el resultado de cruzar una fila de la matriz de vocabulario de un lado a otro y asignarle un número arbitrario al conjunto de palabras obtenido. Dicho de otra forma, un *synset* es un conjunto de palabras de la misma clase morfo-sintáctica que pueden ser intercambiadas en un contexto particular [52]. Los significados se representan mediante *synset* y las relaciones semánticas se simbolizan mediante enlaces entre *synset*.

Un ejemplo de consulta realizada sobre *WordNet* por definiciones y sinónimos para la palabra *sentence* se presenta en la figura 3.6, donde cada categoría léxica para la palabra existen definiciones asociadas a diferentes sentidos, dependiendo del contexto. Adicional a la definición, se indica un ejemplo de uso para la palabra mediante frases. Los sinónimos siguen este mismo formato, es decir, los sinónimos presentados son ubicados según la correspondiente categoría léxica y sentido.

Por otro lado, *Moby Thesaurus* es el *thesaurus* más grande en inglés, inicialmente de uso privativo y luego liberado. Contiene alrededor de dos millones y medio de sinónimos y términos relacionados. El formato de los resultados presentados por esta herramienta está en ascii.[10] y la figura 3.7 presenta una consulta realizada para la palabra *car*.

[8] Listado de palabras, términos o temas relacionados entre sí en forma jerárquica [72].
[9] Diccionario en el que se registran las palabras que conoce un hablante [74].
[10] Del inglés *American Standard Code of Information Interchange*.

Overview of noun sentence

The noun sentence has 3 senses (first 3 from tagged texts)

1. (31) sentence −− (a string of words satisfying the grammatical rules of a language; "he always spoke in grammatical sentences")
2. (5) conviction, judgment of conviction, condemnation, sentence −− ((criminal law) a final judgment of guilty in a criminal case and the punishment that is imposed; "the conviction came as no surprise")
3. (3) prison term, sentence, time −− (the period of time a prisoner is imprisoned; "he served a prison term of 15 months"; "his sentence was 5 to 10 years"; "he is doing time in the county jail")

Overview of verb sentence The verb sentence has 1 sense (first 1 from tagged texts)

1. (6) sentence, condemn, doom −− (pronounce a sentence on (somebody) in a court of law; "He was condemned to ten years in prison")

Synonyms/Hypernyms (Ordered by Estimated Frequency) of noun sentence 3 senses of sentence

Sense 1
sentence
=> string of words, word string, linguistic string

Sense 2
conviction, judgment of conviction, condemnation, sentence
=> final judgment, final decision

Sense 3
prison term, sentence, time
=> term

Synonyms/Hypernyms (Ordered by Estimated Frequency) of verb sentence

1 sense of sentence

Sense 1
sentence, condemn, doom
=> declare

Figura 3.6: *Un ejemplo de salida de WordNet.*

Respecto a la estrategia inicial de producción de textos, el papel de las bases de datos léxicas, *WordNet* y *Moby Thesaurus* es el empleo de sus sinónimos. Éstos sustituyen cada término original de los textos, y tiene en cuenta las partes de voz asignadas por el *SVM-Tagger*. Inicialmente se buscaron diccionarios de sinónimos de uso libre especializados en el área de electrónica, ante la falta de estos recursos, se emplea *WordNet*, al contar con una organización que permite relacionar determinadas partes de voz del Penn TreeBank con las categorías léxicas de *WordNet*, verbo, sustantivo, adjetivo y adverbio. No todas las palabras pueden ser asociadas a dichas categorías, porque ciertos textos contienen partes de voz como determinantes, modales, pronombres, preposiciones, conjunciones, entre otras. Sin embargo, *WordNet* no produce los suficientes sinónimos, teniendo en cuenta la variedad de palabras

1 definition found

From Moby Thesaurus II by Grady Ward, 1.0 [moby−thesaurus]:

63 Moby Thesaurus words for "car":

Pullman, Pullman car, auto, autocar, automobile, baggage car, boat, boxcar, buggy, bus, caboose, carriage, chair car, coach, coupe, covered waggon, crate, day coach, diner, dinghy, dining car, drawing room, flat, flatcar, gondola, heap, jalopy, limousine, local, luggage van, machine, mail car, mail van, motor, motor vehicle, motorcar, motorized vehicle, palace car, parlor car, passenger car, phaeton, railway car, reefer, refrigerator car, roadster, roomette, runabout, sedan, sleeper, smoker, smoking car, station wagon, stockcar, tank, tender, touring, truck, tub, van, voiture, waggon, wheels, wreck

Figura 3.7: *Un ejemplo de salida de Moby Thesaurus.*

Tabla 3.2: *Obtención de sinónimos.*

Palabra	**Parte de voz**	**WordNet**	**Moby Thesaurus**
In	IN	–	–
this	DT	–	–
paper	NN (sustantivo)	2:composition, report, theme; 3:newspaper; ...	Bible paper, CD, IOU, ...
,	,	–	–
we	PRP	–	–
present	VBP (verbo)	1: show, demo, exhibit, demonstrate; 11: confront, face; ...	accomodate, accord, ...
a	DT	–	–
new	JJ (adjetivo)	4: fresh, novel; 5: raw; ...	actual, afresh, again, ...
tech	NN (sustantivo)	1: technical school	–
.	.	–	–
:	:	:	:

empleadas en el tema de estudio, debido a que su uso es de propósito general. Se opta por emplear *Moby Thesaurus* como fuente adicional de sinónimos.

La tabla 3.2 representa un ejemplo de la información obtenida respecto a los sinónimos de las palabras originales en un texto. Las palabras con las etiquetas de partes de voz asignadas, que tiene su respectiva equivalencia en las categorías léxicas sustantivo, verbo, adjetivo y adverbio, fueron consultadas en *WordNet* y *Moby Thesaurus* para obtener sus sinónimos. Como se ha mencionado, *WordNet* tiene una disposición que permite especificar un sentido para cada palabra, dentro de cada categoría léxica, por ello, la tabla tiene señalado un número para indicar el sentido en el cuál se aplican los sinónimos, pero por razones de simplicidad, en este documento no se muestra la definición en cada sentido. No ocurre lo mismo para *Moby Thesaurus*, al no estar organizado como *WordNet*. Para éste sólo se tienen en cuenta si los sinónimos también se encuentran en *WordNet* y dentro de la misma categoría léxica, ya que no hace tal distinción, filtrando sinónimos no significativos desde el punto de vista léxico e incluso semántico[11].

Según lo planteado hasta el momento en la forma de producir texto, resta realizar la sustitución de las palabras originales en los textos de entrada por sus respectivos sinónimos. Sin embargo, ¿Cuáles de los sinónimos son los más indicados para intercambiar por las respectivas palabras originales?, ¿De qué depende la selección de dichas palabras?,¿Existen técnicas o herramientas que permitan decidir cuales sinónimos son los más adecuados?. La escogencia se puede realizar aleatoriamente pero existe el riesgo de distorsionar el contenido textual y hacerlo incoherente, lo cual representa un caso de ambigüedad semántica.

3.3.4. La ambigüedad semántica

Una cuestión notable al hablar de los sentidos de las palabras, es quién o qué define cuáles son. Existe una rama del procesamiento del lenguaje natural denominada resolución de la ambigüedad semántica (*WSD*[12]), encargada de decidir el sentido correcto de una palabra

[11] Es decir, palabras sin sentido en un determinado contexto.
[12] Del inglés *Word Sense Disambiguation*.

Tabla 3.3: *Empleo del SenseRelate para corpus no anotado.*

Palabra	**Categoría**	**Sentido**	**Definición**
Here	sustantivo	1	the present location; this place;
there	sustantivo	1	a location other than here; that place;
are	verbo	1	have the quality of being;
three	sustantivo	1	the cardinal number that is the sum of one and one and one
stages	sustantivo	1	any distinct time period in a sequence of events;
:	–	–	–
text	sustantivo	1	the words of something written;
planning	sustantivo	2	the act or process of drawing up plans or layouts for some project or enterprise
,	–	–	–
sentence	sustantivo	1	a string of words satisfying the grammatical rules of a language;
planning	sustantivo	3	the cognitive process of thinking about what you will do in the event of something happening;
and	–	–	–
text	sustantivo	1	the words of something written;
realisation	sustantivo	3	a sale in order to obtain money (as a sale of stock or a sale of the estate of a bankrupt person) or the money so obtained
.	–	–	–

Tabla 3.4: *Empleo del SenseRelate para corpus anotado.*

Palabra	**Parte de voz**	**Categoría**	**Sentido**	**Definición**
Here	RB	adverbio	1	in or at this place; where the speaker or writer is
there	EX	–	–	–
are	VBP	verbo	1	have the quality of being;
three	CD	adjetivo	1	being one more than two
stages	NNS	sustantivo	1	any distinct time period in a sequence of events;
:	:	–	–	–
text	NN	sustantivo	1	the words of something written;
planning	NN	sustantivo	2	the act or process of drawing up plans or layouts for some project or enterprise
,	,	–	–	–
sentence	NN	sustantivo	1	a string of words satisfying the grammatical rules of a language;
planning	NN	sustantivo	3	the cognitive process of thinking about what you will do in the event of something happening;
and	CC	–	–	–
text	NN	sustantivo	1	the words of something written;
realisation	NN	sustantivo	5	making real or giving the appearance of reality
.	.	–	–	–

ante un contexto concreto [75]. La desambiguación no es un fin, sino una labor intermedia necesaria en algunas situaciones del *PLN*. Normalmente los sentidos posibles de una palabra se definen en un repositorio de sentidos, siendo *WordNet* la herramienta más completa en este aspecto.

La desambiguación es esencial en aplicaciones de comprensión de lenguaje. Además, tiene aplicaciones en traducción automática, para determinar la interpretación más adecuada de una palabra en otro idioma; en recuperación de información, donde se pueden eliminar documentos cuando el sentido de una palabra no es relevante; en procesamiento de textos y voz para corregir acentos y pronunciaciones; entre otras.

Una herramienta para realizar desambiguación semántica es *SenseRelate* [76], la cuál usa medidas de relación y similaridad entre palabras en *WordNet*; posee dos algoritmos, uno asigna sentido a cada palabra dentro de un texto (*AllWord*) y el otro ubica el sentido de determinada palabra (*TargetWord*) por contexto. Existe un tercer componente capaz de seleccionar el sentido de una palabra más relacionada con un determinado conjunto de palabras (*WordToSet*). Para llevar a cabo la desambiguación, se calcula un índice de similitud semántica entre una palabra y sus vecinos, a partir del cuál se asigna el sentido más relacionado con ellos.

La tablas 3.3 y 3.4 presentan ejemplos del uso de la herramienta *AllWord* del *SenseRelate*. Pueden tomarse textos anotados con partes de voz como textos sin anotaciones, pero realizar el proceso de desambiguación en textos con sus partes de voz, puede mejorar el desempeño de la herramienta en términos de velocidad y precisión, al reducir el número de palabras a considerar, dado el caso que pertenezca a más de una categoría léxica en *WordNet*.

El SenseRelate AllWord es empleado para seleccionar el sentido adecuado para los conjuntos de sinónimos extraídos de WordNet como posibles términos equivalentes en los textos originales. Ésto refina la sustitución hecha en el procedimiento básico de generación planteado hasta el momento, garantizando un mejor criterio para la selección de las palabras. No obstante, las selecciones siguen estando ligados a la subjetividad del lector, que puede seguir considerando inapropiadas las palabras.

Ya se tienen los criterios suficientes para la implementación de un prototipo de software para generar texto automáticamente. Además, se ponen en evidencia algunos conceptos y problemas clásicos planteados en el estudio de la lingüística computacional. La siguiente sección es la descripción del prototipo base del generador automático de texto desarrollado en el presente trabajo.

3.4. Prototipo inicial propuesto, pruebas y resultados

Originalmente se concibe la creación de un generador automático de textos similar a *SciGEN*, tratando el problema de la coherencia textual con estructuras retóricas. Sin embargo, el estudio de la teoría de la estructura retórica es extenso y se sale de los limites de presente trabajo. Alternativamente se encuentran recursos lingüísticos relacionados con generación de lenguaje natural, los cuales influyen en un nuevo enfoque: partir de artículos editados por humanos y generar textos similares a éstos, manteniendo la estructura gramatical pero cambiando las palabras originales por sinónimos.

El esquema del primer prototipo se muestra en la figura 3.8, cuya implementación fue realizada en *PERL*[13]. Se selecciona un párrafo de un artículo técnico del tema de estudio, se separa por palabras y símbolos ubicándolos línea por línea y se pasa al *SVMTagger* para asignar la correspondiente parte de voz, posteriormente las palabras y sus partes de voz son procesados

[13] *Practical Extraction and Report Language. www.perl.org.*

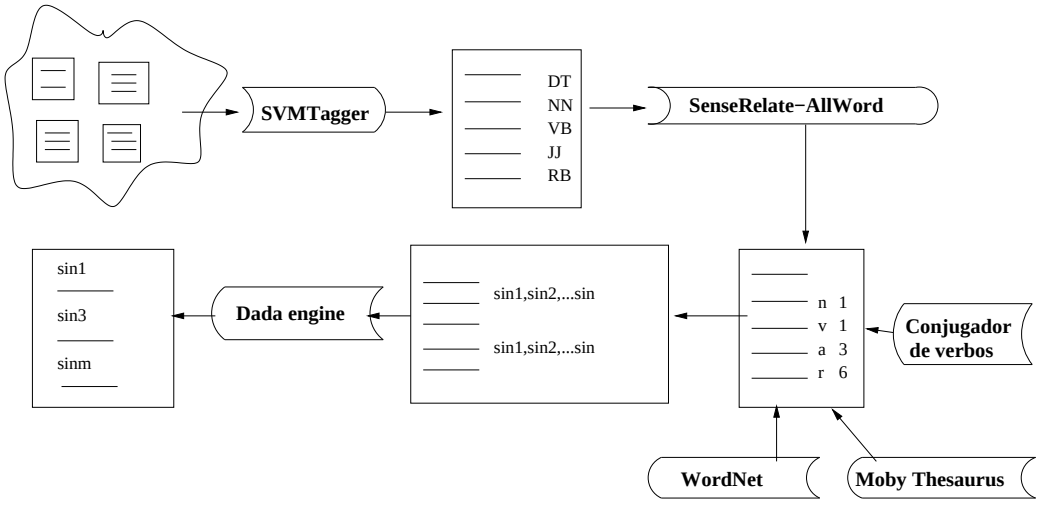


Figura 3.8: *Primer Prototipo.*

```
here three phase text planning , sentence planning text realization
Here three stages : text planning , sentence planning and text actualization
here there_are three stages : textual matter planning sentence planning and text realization
here iii stages : text planning sentence planning and text realisation
Here there_are three stage : text planning , sentence provision and textual matter realization
Here there_are iii stage : text planning , sentence preparation and text actualisation
Here iii stages : textual matter planning sentence provision and textual matter actualisation
Here there_are iii stage : text planning sentence planning and text realization
```

Figura 3.9: *Ejemplo de un texto generado por el primer prototipo a partir de la tabla 3.4*

por el *SenseRelate AllWord* para ubicarlas, de ser posible, en una categoría léxica equivalente en *WordNet* junto con el sentido más apropiado al contexto del escrito. Partiendo de esta información, es posible realizar la consulta de sinónimos y palabras relacionadas a *WordNet* y *Moby Thesaurus*[14]. Toda la información hasta este momento es expresada en términos de gramáticas regulares, una por cada oración del texto, que pueden tomarse como entrada por el *Dada Engine*, para generar como salida la oración, reemplazando las palabras originales por sus respectivos sinónimos. La figura 3.9 muestra un ejemplo del texto generado para la frase: *"Here there are three stages: text planning, sentence planning and text realization."* por medio del proceso en este prototipo.

Los textos resultantes de las pruebas realizadas presentan inconsistencias en su contenido. Dichas inconsistencias no hacen posible ver al texto producido con calidad en términos de fluidez y coherencia, haciendo notar las fallas en las herramientas empleadas en el proceso de generación del prototipo. Los factores causantes están relacionados con el etiquetado de partes de voz, los recursos lingüísticos utilizados como fuente para obtener los sinónimos y el modo de realizar el texto final.

Para el caso del etiquetador de partes de voz, se encuentran problemas vinculados con el comportamiento de la herramienta frente a formas textuales y su desempeño. Una dificultad observada es la no distinción entre verbos y algunas expresiones usadas como auxiliares en el idioma inglés, permitíendo, por ejemplo, asignar sinónimos a modales cuando estos no tie-

[14]Los sinónimos seleccionados de *Moby-Thesaurus* también deben estar en *WordNet* dentro la misma categoría léxica de la palabra original.

nen. También se presentan inconvenientes con expresiones agrupadas: el *SVMTagger*, en su modelo de aprendizaje, no detecta automáticamente este tipo de formas que tienen diferente significado cuando son agrupadas y cuando están por separado. Por ejemplo, la expresión *on the other hand*, tiene un significado distinto al de cada uno de sus componentes. Debido a que el *SVMTagger* las toma por separado, los sinónimos generados distorsionan notablemente el significado del texto. Por otro lado, al aparecer una palabra que no esté en el modelo de aprendizaje, siempre asigna la etiqueta de sustantivo (NN); posiblemente esa palabra desconocida no sea sustantivo y los sinónimos asignados son inadecuados, ahondando más el problema de la ambigüedad. Otra dificultad relacionada es la baja velocidad de ejecución del etiquetador que en general demora el proceso de generación. Finalmente los parámetros de ajuste de la herramienta pueden influir en la precisión de las partes de voz.

Los autores del presente trabajo realizaron una adaptación en el programa con el objetivo de mejorar la precisión del etiquetador aclarando que no se intervino directamente en la herramienta SVMTagger. La modificación consistió en tomar todas las formas de los verbos auxiliares (*do, can, would, will be, have been entre otros*) y las expresiones agrupadas (*there is, how much, on the other hand entre otros*) y asociarlas en una nueva parte de voz denominada *FIX* solucionándose el inconveniente de los sinónimos en este tipo de palabras que en realidad no lo necesitan. Sin embargo, con las términos técnicos que el etiquetador no reconocía y asignaba siempre la categoría léxica de sustantivo no se pudo hallar una solución viable pues se necesitaba tener un *corpus* anotado y representativo del inglés con ese tipo de palabras para volver a entrenar el modelo de aprendizaje *SVMTlearn*.

En *WordNet* y *Moby Thesaurus* existen consideraciones respecto al tratamiento de los verbos y su naturaleza. Los sinónimos mostrados por los diccionarios para los verbos en un tiempo distinto al presente carecían de conjugación: sólo se muestran en su forma base. Entonces se hizo necesario implementar un conjugador de verbos debido a la ausencia de una herramienta de código abierto de este tipo adaptable en el prototipo. Estos recursos lingüísticos son de caracter general, lo cual es poco útil con relación a los términos particulares empleados en la jerga del tema sobre el cual son generados los textos, y dificulta tener un número considerable de sinónimos apropiados para ser remplazados por las palabras originales. A su vez, *Moby thesaurus* muestra toda la información sin clasificarla en categorias léxicas y con un gran número de términos relacionados no acordes al contexto, filtrando sinónimos inapropiados. Adicionalmente, debe tenerse en cuenta la desambiguación semántica de sinónimos en una tarea independiente, para seleccionar el sentido y los sinónimos más apropiados al contexto de las frases.

El modo de realizar el texto por medio de las gramáticas libres de contexto ingresadas al *Dada Engine* es esencialmente estático lo que dá poca flexibilidad y fluidez a la salida final de texto, con relación a las entradas del prototipo. El procedimiento es muy básico, consiste simplemente en reemplazar palabras ya existentes por los sinónimos correspondientes extraídos de los recursos lingüísticos empleados para este fin, pero la estructura gramatical de la oración, representada en las secuencias de partes de voz, permanece sin modificación, lo cual no hace notable la impresión de un texto bien formado, en términos de coherencia.

En vista de estos resultados, se concluye que no es factible el desarrollo de una herramienta de generación automática de textos acorde a las expectativas iniciales. En gran medida, la baja calidad de los textos generados se atribuye a las características de los componentes de este prototipo, especialmente a los recursos lingüísticos (etiquetador de partes de voz y bases de datos léxicas) y a las falencias derivadas por la realización del texto. No obstante, este prototipo es esencial para detectar las implicaciones en el proceso global de generación y establecer así los límites y alcances de este trabajo.

En general, se determina la necesidad del ser humano como apoyo durante las tareas intermedias de generación, esencialmente dónde su conocimiento es fundamental. Permitir a los componentes del sistema de GLN realizar este proceso puede influir notablemente en la calidad de los textos producidos, considerando que el estilo de edición de textos de cada persona es particular, especialmente en campos de aplicación donde el conocimiento es técnico. Los recursos lingüísticos utilizados en tareas intermedias de la generación, deben ser construidos acordes al contexto de la aplicación para generar contenidos textuales similares a los editados por humanos y adicionalmente evitar problemas de coherencia y ambigüedad. Por otro lado, es necesario replantear la manera de hacer la realización final del texto, buscando más aleatoriedad y fluidez, lo cual motiva a enfocar esfuerzos hacia la construcción de un *corpus* anotado de texto con apoyo del etiquetador de parte de voz, las bases de datos léxicas (*WordNet* y *Moby Thesaurus*), el ser humano y el generador de oraciones empleando técnicas que fundamenten estrategias de formación automática de texto.

Capítulo 4

Sistema propuesto

El sistema generador de textos propuesto en este trabajo se desarrolla con base a los resultados obtenidos de la implementación experimental descrita en el capítulo anterior. Dichos resultados se enfocan hacia el diseño de un procedimiento para la construcción de un *corpus* lingüístico que se describe en la primera sección. Con base en éste, se plantean dos estrategias de generación de oraciones teniendo en cuenta la similitud semántica como criterio de la calidad para las nuevas oraciones. Adicionalmente, se desarrolla un prototipo de herramienta computacional para validar las pruebas realizadas a estos planteamientos aspectos tratados en la siguiente sección.

4.1. La construcción del *corpus* lingüístico

El *corpus* de texto es una colección de escritos con una estructura definida acorde a las necesidades para la cual se utiliza [62]. La organización planteada para el *corpus* desarrollado, consta de entidades denominadas *títulos* representadas por una frase o enunciado, que en teoría sugiere el tema del contenido textual y se compone de *párrafos*. El esquema de los *párrafos* se restringe a especificaciones sobre el tipo de contexto y la descripción de su función dentro del *título*. Por ejemplo, un esquema de *párrafo* puede llamarse "Introducción" y se describe como "la explicación global del contenido de un documento". La figura 4.1 representa el formato de dichas unidades de *corpus*.

Cada *oración* tiene asociada una denominación, una nota descriptiva y un número que indica su posición dentro de determinado esquema para párrafo, es decir, una *oración* denominada "idea principal" se describe como "el resumen el tema tratado en el documento" y se ubica en la primera posición dentro del párrafo de tipo "Introducción". Las oraciones deben estar asociadas a un sólo esquema para párrafos y no pueden existir como elementos aislados.

Comúnmente, la oración se define como la unidad mínima de comunicación con significado completo y está formada por palabras[1]. Sin embargo, para el desarrollo del presente trabajo, no es útil considerar solamente las palabras como única información explícita en el contenido de las oraciones, es por ello que cada frase dentro del *corpus* tienen adicionalmente la notación de la categoría gramatical de cada palabra (parte de voz) y un listado respectivo de otras palabras relacionadas semánticamente (sinónimos). La palabra, su parte de voz, sus sinónimos y la posición dentro de la oración, hacen parte de la unidad conceptual denominada *término* y no es posible concebir un *término* sin tener relación alguna con una oración.

[1] La organización de las palabras se rige por numerosas reglas gramaticales.

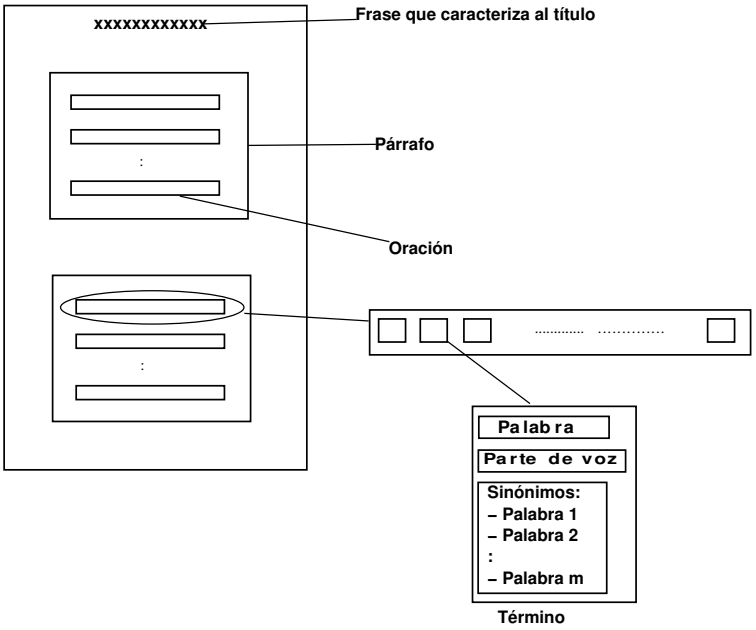


Figura 4.1: *Unidad de corpus de texto.*

En general, los cimientos del *corpus* lingüístico son los *títulos*, conformados por *párrafos* que a su vez poseen información relativa a su función dentro del escrito. Así mismo, cada *oración* dentro de un *párrafo*, contiene una anotación que señala la posición y su significado dentro de éste, representando un contexto determinado. Un conjunto de *términos* constituyen una *oración* y se encuentran organizados de acuerdo a un orden. El *corpus* no sólamente posee anotaciones lingüísticas conformadas por la parte de voz y los sinónimos de cada *término*, también conceptuales indicado por medio de notas. Se considera un *corpus* como anotado si posee el componente lingüístico.

A partir de la definición de *corpus* expuesta anteriormente, se plantea el procedimiento a seguir para conformar una colección de *títulos*. La figura 4.2 presenta el esquema general de este procedimiento y consta de los siguientes pasos:

1. **Seleccionar los textos que constituyen las fuentes de contenido**. Este paso está sujeto a criterio de los expertos, los cuales definen el tema a representar con el *corpus*. A cada escrito se le asigna un enunciado representativo, el cuál caracteriza a cada *título*.

2. **Formar las especificaciones de párrafos posibles a contener en los títulos**. Para cada una se indica su tipo y descripción; de la misma manera para las *oraciones* con su posición dentro del *párrafo*. Esta labor corresponde a los expertos, quienes deben considerar los patrones de escritura detectados en los textos escogidos. De esta manera, se crean las plantillas de *párrafos*.

3. **Adaptar cada texto a las plantillas de párrafo**. Esto también es una tarea que involucra la subjetividad de los expertos. Puede ser difícil o incluso imposible realizar esta adaptación de escritos; entonces se deben realizar ajustes a las plantillas para párrafos o buscar otros textos adaptables a las plantillas. En este paso, los *títulos* poseen anotaciones de tipo conceptual.

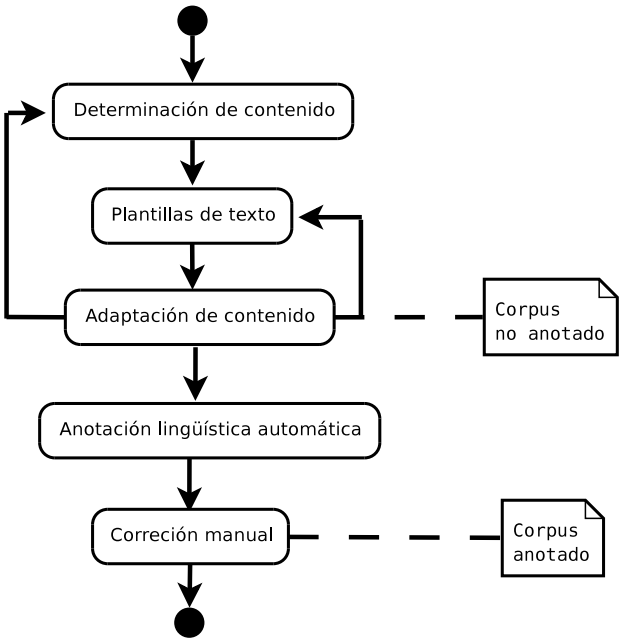


Figura 4.2: *Procedimiento para la construcción de corpus.*

4. **Agregar anotaciones lingüísticas a los títulos**. Se pasa cada uno por el etiquetador automático de partes de voz (*SVMTagger*), obteniendo así marcas gramaticales para las palabras, de esta forma, se agiliza el tiempo de construcción del *corpus* anotado.

5. **Evaluar los títulos con sus partes de voz**. Cada *título* se pone a consideración de los expertos para revisar, corregir y aprobar dichas marcas gramaticales. Además, de ser posible, coloca para cada palabra el conjunto de sinónimos. Así, se tiene un *corpus* anotado conceptual y lingüísticamente.

Se plantea la hipótesis según la cual, la calidad de las frases generadas depende de las *oraciones* anotadas en el *corpus*. Entre ellas pueden establecerse relaciones de similitud semántica, dependiendo de los términos relacionados contenidos en sus palabras y sus marcas gramaticales ayudan en la formación de la estructura sintáctica para las nuevas oraciones. Así, las estrategias de generación de frases toman como entrada oraciones anotadas lingüísticamente en el mismo contexto, para producir una oración semánticamente similar y con una estructura gramatical distinta.

4.2. La generación de frases

El diagrama de la figura 4.3 representa el modelo propuesto para formar una oración automáticamente. Para producir oraciones se parte de la información suministrada por el *corpus* anotado lingüísticamente escogiendo una oración objetivo, luego se obtiene una determinada cantidad de frases en el mismo contexto y se toman como entrada para la estrategia de generación empleada, la cuál produce un conjunto de oraciones pre-generadas. Por último se considera la similitud semántica de la oración objetivo con dicho conjunto, seleccionando la de mayor puntuación.

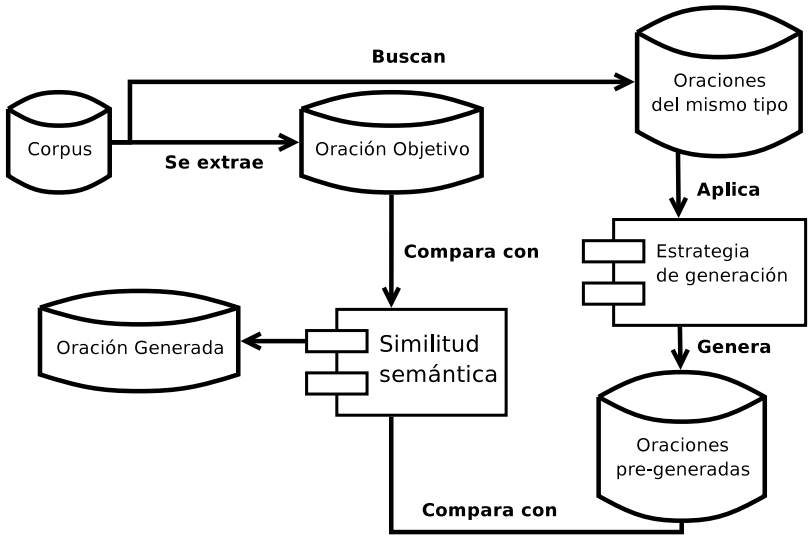


Figura 4.3: *La generación de frases.*

4.2.1. La similitud semántica

La mayoría de los motores de búsqueda implementan el modelo del espacio vectorial[2] como estructura de datos y criterio de alineamiento[3] en función del parecido de una pregunta con documentos almacenados [78]. El modelo del espacio vectorial es un método diseñado para recuperación de información donde se intenta recoger la relación de un documento Di representado como de un vector de términos, en una colección de N documentos, que poseen m características.

$D_i -> \vec{d_i} = (c_{i1}, c_{i2}, ...c_{im})$; donde $\vec{d_i}$ identifica el grado de aceptación del documento D_i en cada una de las m características. En éste vector, c_{ik} expresa la frecuencia de D_i sobre la característica k; el concepto "característica" se refiere a la ocurrencia de determinadas palabras o términos en el documento, aunque no siempre es el único aspecto a considerar [63].

El proceso a seguir consiste en seleccionar aquellos términos útiles que permitan discriminar unos documentos de otros. No todas las palabras contribuyen con la misma importancia en la caracterización del documento, desde el punto de vista de la recuperación de información existen palabras casi vacías de contenido semántico, como artículos, preposiciones o conjunciones, poco útiles en el proceso. Una vez seleccionado el conjunto de términos representativos, es necesario obtener el valor de cada elemento en el vector del documento con el objeto de determinar la capacidad de representación de un término para un documento dado calculando su número de apariciones y obteniendo la frecuencia del término en el documento.

El modo más simple de calcular la similitud entre dos o más documentos, es realizar el producto escalar de los vectores que los representan. El índice de similitud más utilizado es el coseno del ángulo formado por ambos vectores.

$$cos(D_1, D_2) = \frac{\vec{D_1} * \vec{D_2}}{|\vec{D_1}| * |\vec{D_2}|}.$$

Donde $|D_1|$ es la dimensión del vector $\vec{D_1}$ calculado con la forma de distancia euclidiana

[2]También conocido como modelo vectorial.

[3]Arreglo de secuencias, que muestran coincidencias y diferencias entre ellas. Tomado de fbio.uh.cu/bioinfo/glosario.html.

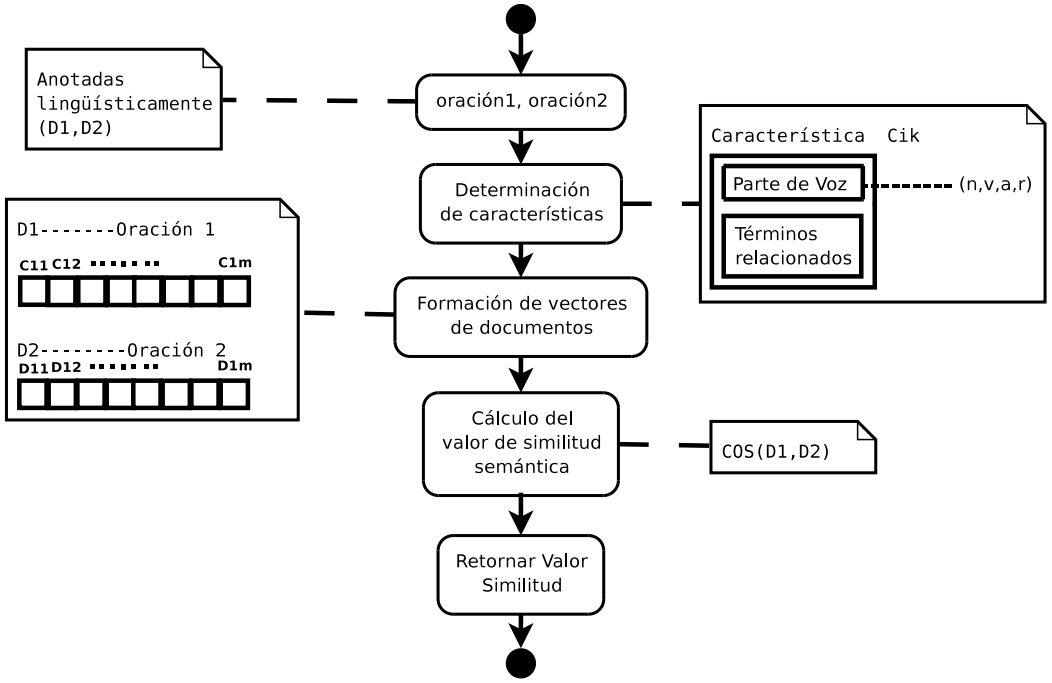


Figura 4.4: *Cálculo del valor de similitud entre dos oraciones.*

Tabla 4.1: *Oraciones de ejemplo.*

Documento 1	The	nature	is	precious	,	help	me	to	preserve	it	.
	DT	NN	VBZ	JJ	,	VB	PRP	TO	VB	PRP	.
				pretty		aid			protect		
Documento 2	Nature	is	life	,	let	treats	protect	her	.		
	NN	VBZ	NN	,	VB	VBZ	VB	PRP	.		
	earth						conserve				

$|D_1| = \sqrt{c_{i1}^2 + c_{i2}^2 + ... + c_{im}^2}$. Si el cálculo está muy cercano a uno significa que las dos frases tienen un alto grado de similitud, el caso opuesto es si da proximo a cero [64].

El modelo vectorial y la similitud son tomados como base para la selección de oraciones siendo un procedimiento intermedio en la generación de texto. El diagrama de flujo de la figura 4.4 muestra la serie de pasos para determinar el valor de similitud semántica entre dos oraciones del *corpus*.

1. Partir de las dos oraciones, consideradas como los documentos. Se toman como ejemplo las frases mostradas en la tabla 4.1.

2. Determinar las características relevantes de los dos documentos. Se consideran relevantes los términos etiquetados para las categorías léxicas verbo, sustantivo, adjetivo y adverbio, por su valor semántico y gramatical. Cada caracteristica contiene además, un grupo conformado por las palabras y sinónimos de los términos comunes dentro de las categorías léxicas. Por ejemplo, con los términos *(preserve—VB—protect,conserve)* y *(protect—VB—conserve)* se puede formar la característica (v—preserve,protect,conserve) ya que están en la misma categoría léxica y contienen palabras en común. En el caso del ejemplo ilustrativo, se conforman las siguientes características, ver figura 4.5.

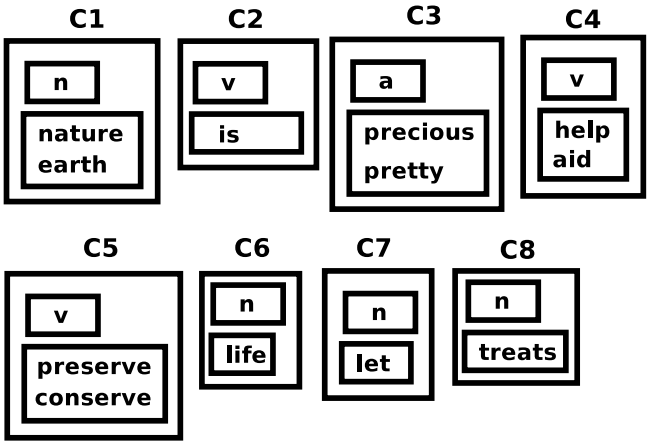


Figura 4.5: *Características para las oraciones de ejemplo.*

Tabla 4.2: *Características para las oraciones de ejemplo.*

	C1	**C2**	**C3**	**C4**	**C5**	**C6**	**C7**	**C8**
Documento 1	1	1	1	1	1	0	0	0
Documento 2	1	1	0	0	1	1	1	1

3. Se forman los vectores de documento. Se calcula la frecuencia de aparición de cada característica en los dos documentos, considerando la equivalencia en las partes de voz y las relaciones entre palabras y sinónimos entre los términos de los documentos. Los resultados son los vectores de las dos oraciones y pueden verse en la tabla 4.2, para el caso del ejemplo ilustrativo.

4. Finalmente, se obtiene el valor de similitud, empleando el coseno del ángulo entre los vectores

$$\begin{aligned} |D_1| &= \sqrt{5} \\ |D_2| &= \sqrt{6} \\ \vec{D_1} * \vec{D_2} &= 3 \\ cos(D_1, D_2) &= \frac{\vec{D_1} * \vec{D_2}}{|\vec{D_1}| * |\vec{D_2}|} = \frac{3}{\sqrt{5} * \sqrt{6}} = 0{,}5477. \end{aligned}$$

Este valor determina el grado de similitud entre las dos frases.

4.2.2. Primera estrategia

El principio de esta estrategia se basa en la búsqueda preferente por profundidad, la cual selecciona un camino a seguir, aumentando los niveles de profundidad hasta llegar a una posible solución o final del camino. En caso de alcanzar el final del camino se revierte la busqueda y se expanden los nodos de niveles menos profundos [65].

La tabla 4.3 y la figura 4.6 muestran un conjunto de tres oraciones a las cuales se va a aplicar la estrategia, descrita en los siguientes pasos:

1. **Seleccionar la palabra inicial**: consiste en tomar los primeros términos de manera única en las oraciones del conjunto de entrada. Un término es distinto de otro si difieren

Tabla 4.3: *Oraciones de ejemplo*

Objetivo	In	section	2	,	the	circuit	typology	will	be	described
	IN	NN	CD	,	DT	NN	NN	MD	VB	VBN
							classification, categoriza-tion			
Oración 1	In	section	II	,	the	model	classification	is	derived	
	IN	NN	CD	,	DT	NN	NN	VBZ	VBN	
							typology			
Oración 2	Section	V	draws	some	conclusions					
	NN	CD	VBZ	DT	NNS					

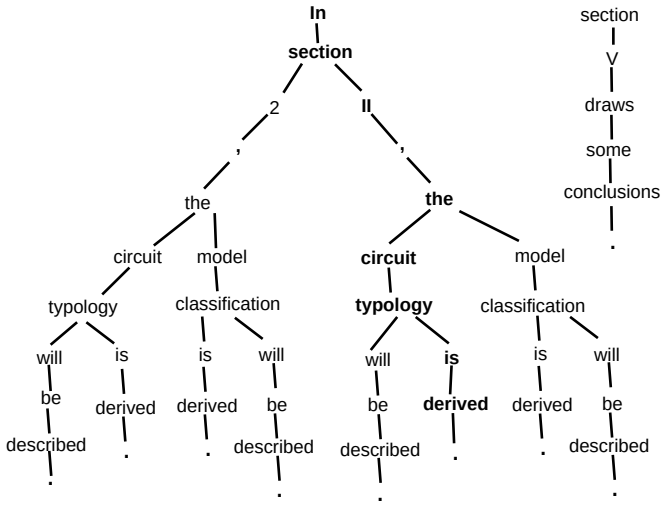


Figura 4.6: *Selección de una oración empleando la primera estrategia*

en su palabra o parte de voz. De estos términos se escoge uno al azar, conformando el término inicial de la nueva frase. Los primeros términos sin repetir son *(In, Section)*, de ellos se escoge aleatoriamente el término *In*.

2. **Generar términos siguientes**: a partir del término generado anteriormente, se buscan repeticiones o sinónimos de éste. Dos términos son sinónimos si coinciden en su parte de voz y al unir sus palabras y conjuntos de sinónimos existen elementos comunes. Cómo resultado de la búsqueda, se obtiene un grupo de nodos cuyas ramas corresponden a las palabras que siguen, para escoger uno de forma aleatoria. En caso de no formar este conjunto, se toma el siguiente término. En el ejemplo, se avanza al siguiente nivel encontrando la posibilidad *(section)*. Para *section* existen dos términos sinónimos relacionados en el mismo nivel *2, II*, escogiéndose uno aleatoriamente *II*, y así sucesivamente para los posteriores niveles.

3. **Criterio de parada**: realizar el paso dos hasta generar como término el punto final. Al darse lo anterior, se ha obtenido una nueva frase. El proceso llega al punto y como resultado se produce la frase *In section II, the circuit typology is derived.*

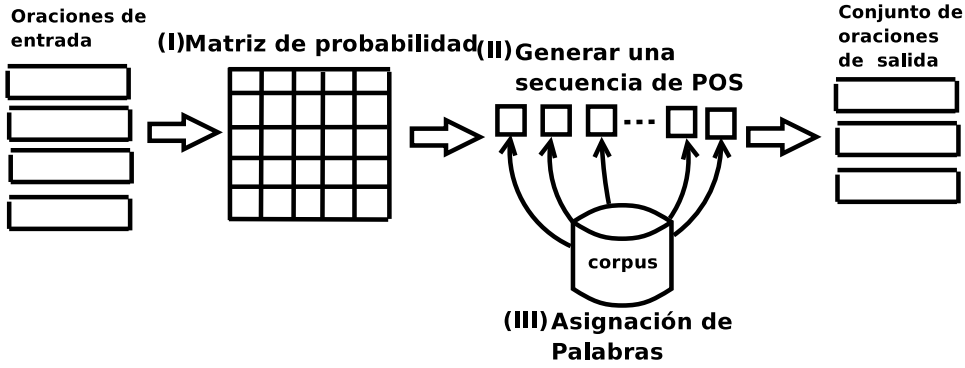


Figura 4.7: *Segunda estrategia de generación* DISGEN.

Tabla 4.4: *Oraciones de ejemplo*

O	In	section	2	,	the	circuit	typology	will	be	described
	IN	NN	CD	,	DT	NN	NN	MD	VB	VBN
S1	In	section	II	,	the	model	classification	is	derived	
	IN	NN	CD	,	DT	NN	NN	VBZ	VBN	
S2	Section	V	draws	some	conclusions					
	NN	CD	VBZ	DT	NNS					

El procedimiento se puede visualizar como un árbol donde cada posibilidad es un camino distinto, para el caso de las tres oraciones ejemplo, se pueden crear hasta nueve oraciones si se combinan los términos iguales y sinónimos. Sin embargo, no siempre es posible crear oraciones, para el caso de la frase *Section V draws some conclusions.* no aparece ninguna posibilidad y por tanto sólo generaría la original.

4.2.3. Segunda Estrategia

El procedimiento consta de tres etapas: cálculo de tabla de probabilidades, generación de secuencia de partes de voz y asignación de palabras, ver figura 4.7.

La descripción detallada de esta estrategia se ilustra a través de las oraciones mostradas en la tabla 4.4. Se asume a *s1* y *s2* como las fuentes para expresar la idea de *0*, las tres se toman como entrada al algoritmo.

1. **Formación de la matriz de probabilidades**: en esta etapa, se obtiene la frecuencia de aparición para determinada etiqueta POS seguida de otra. El resultado para *0*, *S1* y *S2* se muestra en las tabla 4.5, deduciendo por ejemplo, que después de un determinante (DT) sucede dos veces un sustantivo en singular (NN), es decir, el elemento en la fila i y la columna j es el número de veces que la etiqueta POS j ocurre después de la etiqueta POS i. La matriz resultante es cuadrada y siempre tiene un inicio y un fin de oración. El cálculo de las probabilidades se realiza dividiendo la frecuencia de cada elemento una entre la suma de todos los componentes de la fila, ver tabla 4.6.

Tabla 4.5: *Matriz de frecuencias*

	IN	**NN**	**CD**	**MD**	**VBZ**	**,**	**DT**	**NNS**	**VB**	**VBN**	**fin**
inicio	2	1									
IN		1									
NN		2	3	1	1						
CD					1	2					
MD									1		
VBZ							1			1	
,							1				
DT		2						1			
NNS											1
VB										1	
VBN											1

Tabla 4.6: *Matriz de probabilidades*

	IN	**NN**	**CD**	**MD**	**VBZ**	**,**	**DT**	**NNS**	**VB**	**VBN**	**fin**
inicio	0.667	0.333									
IN		1									
NN		0.286	0.428	0.143	0.143						
CD					0.333	0.667					
MD									1		
VBZ							0.5			0.5	
,							1				
DT		0.333						0.667			
NNS											1
VB										1	
VBN											1

2. **Producción de secuencias de etiquetas POS** : la obtención de una secuencia se realiza seleccionando sucesivamente las partes de voz, teniendo en cuenta las probabilidades obtenidas en el paso anterior. Se parte de *inicio* y se escoge la siguiente marca segun la probabilidad de ocurrencia, estando allí se obtiene la siguiente bajo el mismo criterio y así sucesivamente hasta llegar a *fin*. La figura 4.8 representa un grafo de estados con todos los caminos posibles a partir de la probabilidad asociada [65]. En el grafo, cada trayectoria desde *inicio* hasta *fin* se considera una producción de secuencia. Un ejemplo de trayectoria puede ser *inicio-IN-NN-CD-,-DT-NN-NN-VBZ-VBN-fin*, pero en realidad la secuencia se tiene en cuenta sin *inicio* y *fin* pues sólo se usan por conveniencia.

   Una consideración importante en la generación secuencias se relaciona con la longitud. Aunque *fin* es el última etiqueta y puede ser tomada como el criterio de parada, la dimensión de las producciones no siempre coincide con las longitudes de las oraciones originales debido a la naturaleza de la matriz de estados. Para ello, es necesario primero fijar este valor y establecer un ciclo que origine las producciones hasta adquirir una del

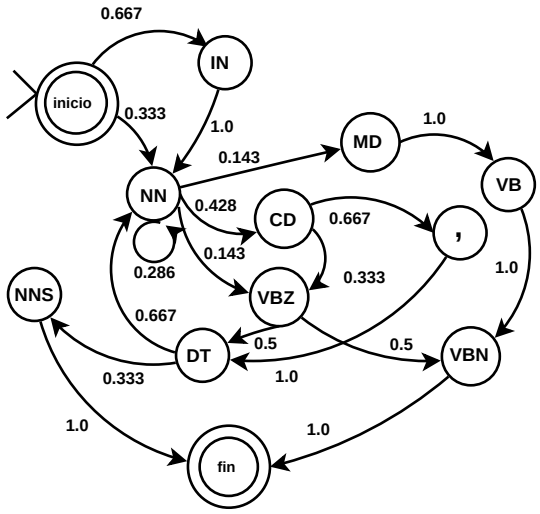


Figura 4.8: *Posibles trayectorias para generar una frase.*

tamaño especificado. El siguiente pseudocódigo describe este ajuste:

```
1. Establecer L (longitud de la secuencia)
2. Secuencia = inicio
3. Hasta que longitud(Secuencia)=L
    secuencia=generarSecuencia(Probabilidades)
   Fin Hasta
```

El valor de L se determina respecto a la distribución de probabilidad de las distintas longitudes de oración en las entradas. Para el ejemplo existen una oración de longitud 9 (*S1*), una de tamaño 10 (O) y otra de 5 (*S2*), lo cual indica que una cada tres veces ocurre la longitud 9. Para el caso del ejemplo ilustrativo, se toma una posible producción y se extraen las correspondientes de O.

```
O=  IN NN CD , DT NN NN MD  VB VBN .
O2= IN NN CD , DT NN NN VBZ VBN .
```

3. **Asignación de palabras:** consiste en localizar las secuencias de términos tomando como base las partes de voz en el mismo orden de las que pueden extraerse de la tira de marcas seleccionada con el paso anterior. Para ello se realizan cuatro pasos, representados en el diagrama de la figura 4.9:

   - Tomar inicialmente toda la secuencia generada y buscar en las oraciones de entrada este patrón de partes de voz.
   - Si se encuentra, se asignan las palabras asociadas a la secuencia, se ubica en la siguiente posición a las ya asignadas y vuelve a realizarse la búsqueda del patrón desde este punto hasta el final de la secuencia. Las palabras pueden ser las originales o alguna dentro del conjunto de sinónimos tomado aleatoriamente.
   - Si no se encuentra el patrón, se toma la secuencia actual sin la última marca y se repite la búsqueda.

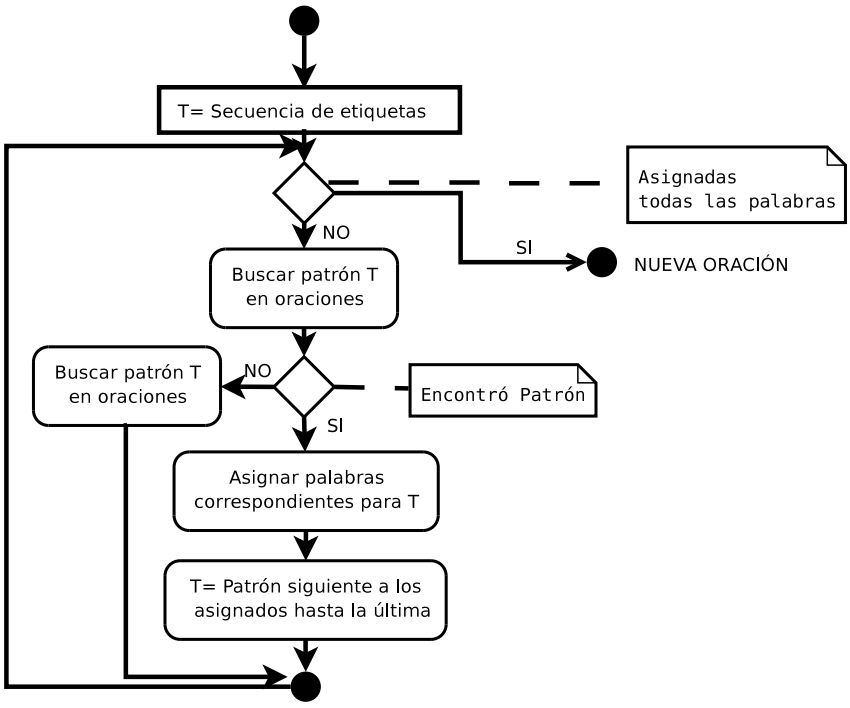


Figura 4.9: *Asignación de palabras.*

Tabla 4.7: *Generación de palabras a partir de las partes de voz.*

Secuencia	**Palabras Seleccionadas**	**Iteración**
IN NN CD , DT NN NN VBZ VBN		1
IN NN CD , DT NN NN VBZ		2
IN NN CD , DT NN NN	In section 2, the circuit typology	3
VBZ VBN	is derived	4

- La asignación termina hasta que ya han sido colocadas todas las palabras.

La tabla 4.7 representa la busqueda necesaria para encontrar las palabras de *02*. En él se indican paso a paso la forma de obtener los conjuntos de palabras hasta llegar a formar la oración. El resultado es *In section 2, the circuit typology is derived*.

4.3. Desarrollo del sistema

En esta sección se presentan los aspectos relacionados con el desarrollo de *DISGEN*. La herramienta *DISGEN* está diseñada para la construcción de *corpus* lingüísticos y realizar la validación de las estrategias de generación de frases planteadas, para el idioma inglés. Sin embargo, no puede considerarse un producto de *software* como tal sino un prototipo que pretende apoyar la edición de textos. Además, se manifiestan los resultados del conocimiento adquirido durante la revisión hecha en este trabajo y el aporte original al estudio del problema. La descripción de este desarrollo se inicia con la metodología utilizada, continúa con el análisis preliminar, el diseño global y detalles de la implementación.

4.3.1. Metodología de desarrollo y análisis del sistema

Para el desarrollo del software propuesto, se aplica la metodología de desarrollo de software conocida como *prototipado evolutivo*, porque se ha mejorado la base de trabajo desarrollada, producida a partir del prototipo inicial y las pruebas realizadas, que llegaron a establecer la necesidad de construir corpus lingüísticos y crear las estrategias de generación de frases (ver sección 3.4). Todo esto se toman como inspiración para implementar la aplicación, cuyo diseño crece a medida que se incluyan nuevos componentes, para cubrir las necesidades de la construcción de *corpus*, las estrategias de generación y los requerimientos de los usuarios del grupo CIDIC, quienes plantearon el problema estudiado a lo largo del presente trabajo.

Utilizando las características de los prototipos incrementales de software para *DISGEN*, se identificaron las funcionalidades básicas mediante un estudio que determinó los actores del sistema y los requisitos generales. Con esta información se elaboran dos diagramas de casos de uso: diagrama de contexto del sistema y diagrama de modelado de requisitos.

Los principales actores identificados en el software *DISGEN* son: invitado, administrador, experto, revisor y editor. En el anexo A se muestra una relación más específica de sus responsabilidades ante el sistema y la información que éste le suministra.

- El invitado es un actor externo al sistema que puede visualizar información relacionada con el proyecto.
- El administrador realiza labores de actualización de registros concernientes a parámetros generales del sistema.
- El actor experto se encarga de la inserción y manejo de los *corpus* de texto y de su anotación lingüística.
- El revisor tiene la responsabilidad de supervisar y aprobar los diferentes textos anotados lingüísticamente por los expertos.
- El editor hace uso del generador de oraciones, cuya salida puede emplear como ayuda al proceso de edición de sus textos.

El diagrama de casos de uso del contexto del sistema destaca los actores necesarios en la vida de la aplicación y los componentes principales en torno al sistema [77]. El diagrama de la figura 4.10 representa dicha vista del sistema para el caso de *DISGEN*. La descripción resumida para cada caso de uso se presenta como sigue:

- *Información del proyecto*: presentación de la documentación teórica relacionada con el proyecto.
- *Usuarios del sistema*: gestión de las cuentas de usuario de los actores que interactúan con el sistema.
- *Esquemas de párrafos*: administración de los esquemas para párrafos que pueden ser construídos en los *corpus* de texto.
- *Corpus lingüístico*: recursos textuales almacenados y empleados por el sistema, que pueden contener anotaciones lingüísticas.
- *Generador de oraciones*: componente que ejecuta los procedimientos relacionados con el proceso de producción automática de frases.

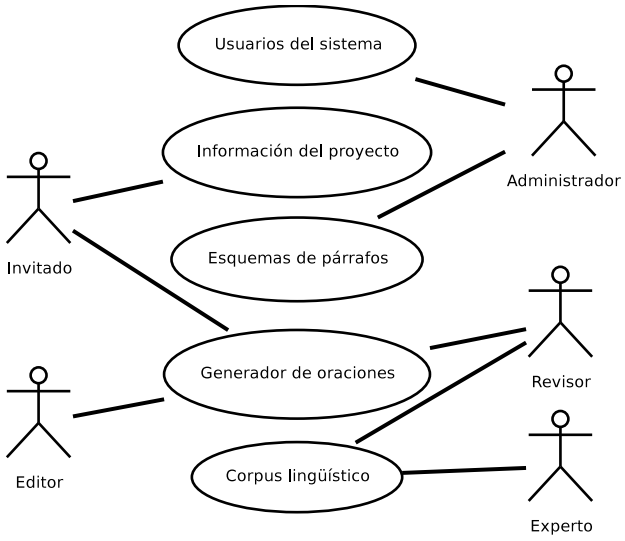


Figura 4.10: *Diagrama de contexto del sistema DISGEN.*

Los casos de uso de modelado de requisitos despliegan la mayoría de las funcionalidades del sistema, con base en el comportamiento que cada actor espera de éste [77]. El diagrama respectivo determinado al final del análisis se incluye en la figura 4.11 , y en el anexo A.2 se especifica la descripción en forma textual de cada uno de los casos de uso.

4.3.2. Diseño global

Partiendo de los objetivos propuestos y de los resultados obtenidos en la etapa de análisis del sistema, el software *DISGEN* enfoca su desarrollo y desempeño en cinco módulos funcionales, cada uno de los cuales está asociado al respectivo tipo de usuario del sistema, además los servicios proporcionados se relacionan directamente con el propósito de cada usuario.

- *Módulo del invitado*: se encarga de la presentación del marco teórico del proyecto, en una interfaz gráfica amigable al usuario compuesta de menúes y ventanas de salida con información en textos y diagramas para facilitar la compresión de los conceptos relacionados. Además, visualiza una demostración del generador de oraciones y referencias para contactar a las personas involucradas.

- *Módulo del administrador*: establece los parámetros de funcionamiento del sistema respecto a las cuentas de usuario y los esquemas para párrafos.

- *Módulo del experto*: se encarga de la gestión de la información textual de los *corpus* de texto anotados y no anotados lingüísticamente.

- *Módulo del revisor*: se encarga de dar las funcionalidades necesarias al usuario correspondiente para hacer la revisión de los *corpus* textuales anotados, así como de dar aprobación a los empleados en el proceso de generación.

- *Módulo del editor*: presenta al usuario los párrafos producidos automáticamente, que puede emplear como apoyo para la edición de sus artículos. Además, da la posibilidad de evaluar la calidad de los textos generados.

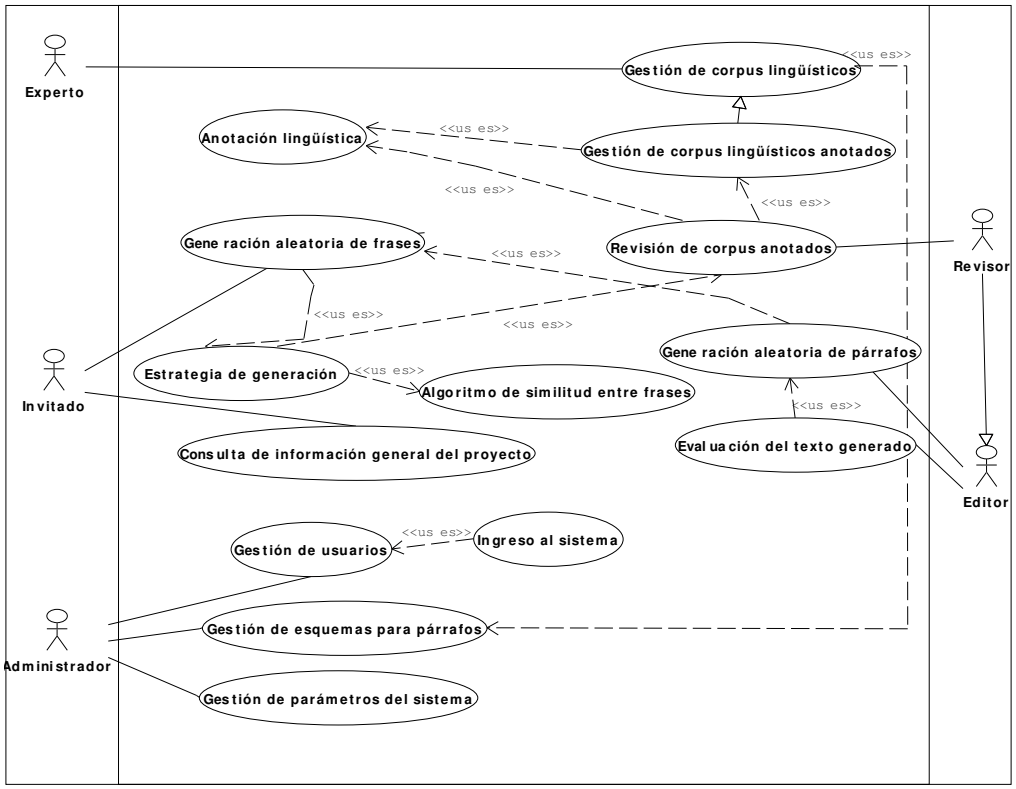


Figura 4.11: *Diagrama de modelado de requisitos para DISGEN.*

En el anexo A.3 se incluyen los diagramas de actividades de las operaciones básicas del sistema.

4.3.3. Implementación

La herramienta tiene como nombre *DISGEN*, agrupada en una única carpeta que contiene directorios organizados según el tipo de recurso que emplea en sistema. Así mismo, esta carpeta contiene una copia de la base de datos de arranque.

El software empleado en el lado del servidor para la implementación de la herramienta fue *PHP 4*, distribuído bajo licencia *PHP* (ver www.php.net) y el sistema administrador de base de datos MySQL 4.1 (ver www.mysql.com). Ambos son software libre.

Para el lado del cliente, los resultados de las peticiones al servidor son mostrados en el lenguaje de marcado *HTML*, también empleado para realizar algunas partes de la interfaz de usuario. Se utilizan también rutinas procedimentales escritas en *JavaScript* para el manejo de eventos del usuario y las hojas de estilo en cascada para especificar formatos en las páginas Web del sistema.

Organización y tecnología de software en la implementación del sistema: la herramienta implica la construcción de una aplicación Web, por lo que la organización que permite una representación de su estructura global es la llamada *arquitectura de tres capas*. En ella, toda aplicación se caracteriza por tener tres tipos de código: código de presentación, código de procesamiento de datos y código de almacenamiento de datos. La principal característica de esta organización es que al ser independientes las capas se disminuye el costo de mantenimiento ya que una modificación en una capa no debe afectar a las demás. Las aplica-

Tabla 4.8: *Arquitectura de tres capas.*

Presentación	**Procesamiento de datos**	**Almacenamiento de datos**
Navegador Web Mozilla Firefox, Internet Explorer, Opera, etc., intérprete de HTML, CSS, JavaScript e imágenes. Residente en el cliente.	Servidor Web Apache, códigos en PHP, Dictd, WordNet y SVMTagger. Residentes en el servidor.	Motor de bases de datos MySQL. Residentes en el servidor.

ciones difieren entre sí por la forma en la cual se organiza el código; en la tabla 4.8 se presenta la especificación de cada una de las tecnologías relacionadas en cada capa en el desarrollo de la herramienta.

- La capa de presentación maneja los aspectos de la interacción con la aplicación. El cliente proporciona el ambiente de esta capa por medio de un navegador, que permite ver los resultados del procesamiento de las peticiones hechas por el cliente en páginas Web. Las tecnologías software empleadas para representar el entorno de esta capa fueron el *HTML*, las hojas de estilo en cascada (*CSS*), códigos en *JavaScript* e imágenes en diversos formatos (la mayoría en *gif y png*), que pueden ser interpretados por la mayoría de navegadores, salvo algunas diferencias que se manejaron con *JavaScript*.

- La capa de procesamiento de datos recibe la entrada de la capa de presentación, interactúa con la capa de almacenamiento de datos, realiza las operaciones definidas en la aplicación y envia el resultado al cliente. El lenguaje de programación empleado para representar esta capa es *PHP*, que se ejecuta a través del servidor Web *Apache*. Este lenguaje tiene muchas características en cuánto a simplicidad de código, conexiones con bases de datos, recursos en el servidor, extensiones con otras aplicaciones y librerías con funciones especializadas. Por medio de los códigos hechos con *PHP*, se accede a *Dict* (usando moby-thesaurus), *WordNet, y SVMTagger*, empleados en el proceso de anotación lingüística del *corpus*.

- La capa de almacenamiento de datos es la encargada de almacenar, recuperar y mantener la información que maneja la aplicación. *MySQL* es el motor de bases de datos del proyecto, empleado como mecanismo principal de almacenamiento. Éste presenta una mejor integración con *PHP*, funciona en sistemas de cliente/servidor, consume muy pocos recursos en procesamiento y memoria y las consultas se hacen con rapidez.

Estructura de los datos de *DISGEN*: en la figura 4.12 se muestra el diagrama de entidad-relación de la base de datos empleada en el proyecto. Seguido, se presenta la descripción de cada una de las tablas (tabla 4.9) y el anexo A.3 provee una descripción detallada de sus campos (diccionario de datos).

Vista de despliegue y componentes: la herramienta implica el uso de Internet y más específicamente del protocolo *HTTP*, para la comunicación de las diferentes partes que la componen. La concepción más detallada del despliegue y los componentes se presenta en la figura 4.13.

- *El servidor*: es el computador que aloja los diferentes archivos en los que está implementado el sistema y ejecuta las instrucciones escritas en los mismos para poder prestar los servicios que el sistema ofrece a los clientes. Dichas instrucciones son ejecutadas y sus resultados servidos al cliente en formato *HTML*, junto con las hojas de estilo en cascada (*CSS*), las secuencias de comandos en *JavaScript* (*JS*), y las imágenes (*IMG*).

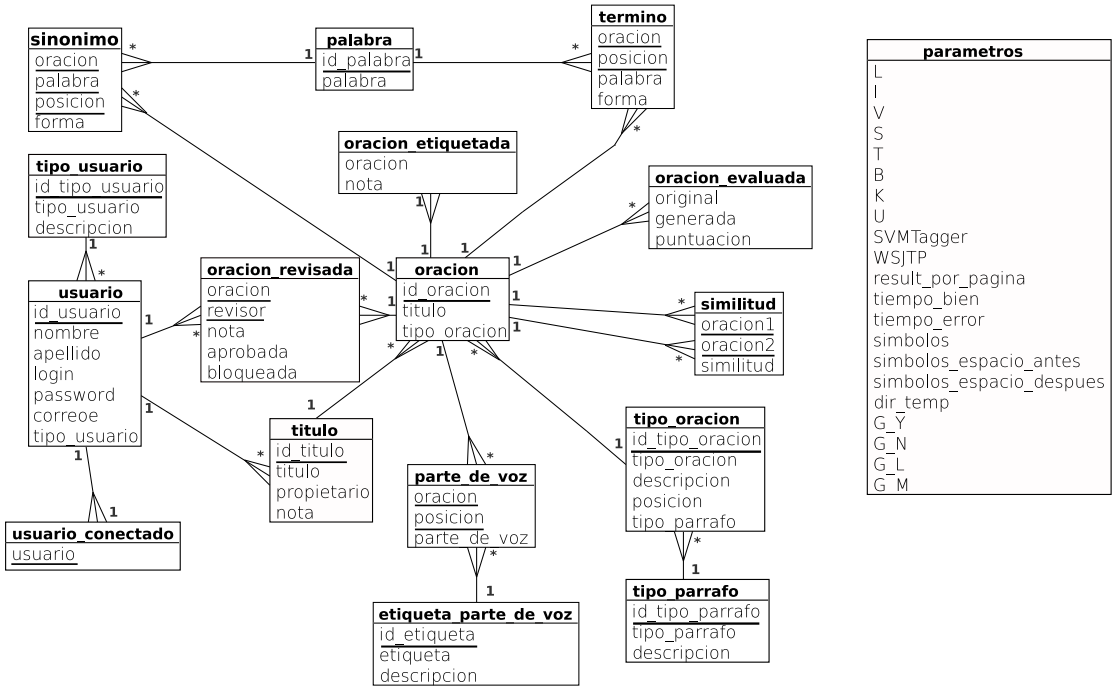


Figura 4.12: *Diagrama de entidad-relación de la base de datos para DISGEN.*

Tabla 4.9: *Descripción de las tablas de la base de datos.*

Nombre de la tabla	**Descripción**
usuario	Usuarios del sistema, tipo administrador, revisores, expertos y editores.
usuario_conectado	Usuarios del sistema que se encuentran activos.
tipo_usuario	Tipos de usuario que el sistema puede tener. Puede ser administrador, experto, revisor y editor.
tipo_parrafo	Especificaciones de los esquemas para párrafos.
tipo_oracion	Especificaciones de los tipos de oración.
etiqueta_parte_de_voz	Partes de voz que pueden asignarse en la anotación lingüística.
titulo	Unidad de *corpus* insertada.
oracion	Oraciones del *corpus*.
termino	Unidades básicas de texto en las oraciones.
palabra	Palabras que están en los *corpus*.
parte_de_voz	Partes de voz que tienen los términos de las oraciones.
sinonimo	Sinónimos de los términos en las oraciones.
oracion_etiquetada	Oraciones que tienen anotaciones lingüísticas.
oracion_revisada	Oraciones que han sido supervisadas por los revisores.
similitud	Valor de la similitud entre pares de oraciones del mismo tipo.
oracion_evaluada	Oraciones que tienen una valoración numérica respecto a su calidad.
parametros	Parámetros generales de funcionamiento del sistema.

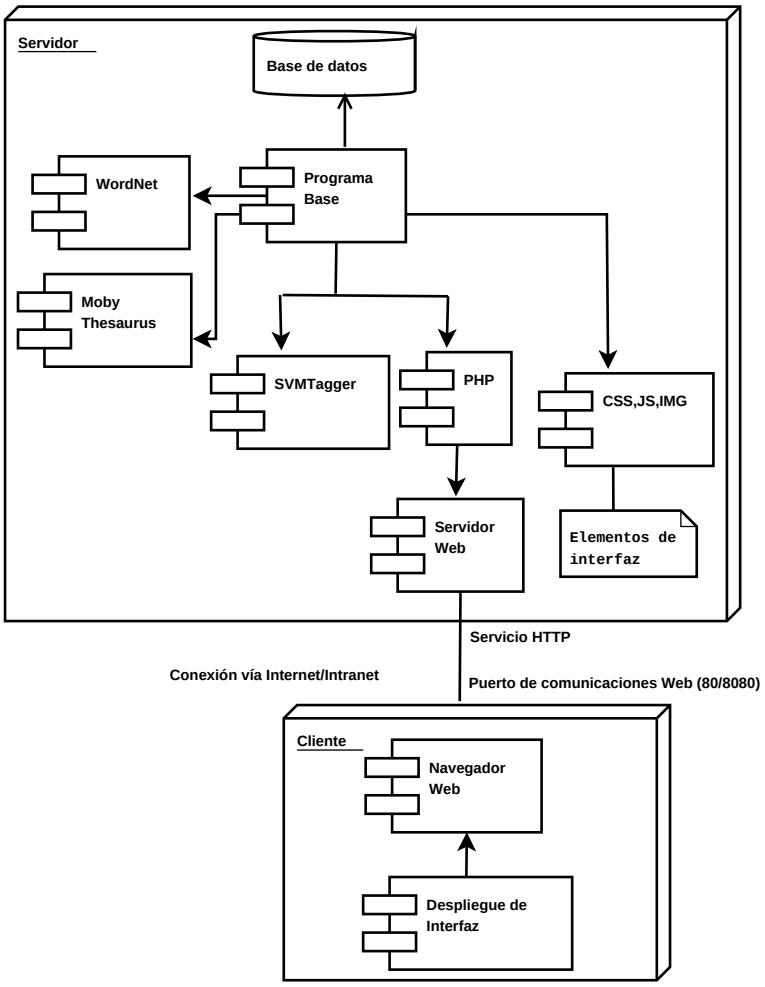


Figura 4.13: *Diagrama combinado de despliegue y componentes para DISGEN.*

De la misma manera, las peticiones de los clientes son transmitidas al servidor a través del protocolo *HTTP* para su ejecución. El servidor también es responsable de alojar y gestionar los datos de la aplicación mediante la ejecución del programa manejador de bases de datos y el manejo de los archivos temporales de la anotación lingüística. También tiene instalados los diccionarios léxicos *(Dictd y WordNet)* y el etiquetador de partes de voz (*SVMTagger*).

- *Los clientes*: son otros sistemas que se conectan al servidor para solicitar servicios de éste. Para que un computador sea cliente de *DISGEN* es necesario que cuente con una conexión a Internet o una comunicación *HTTP* entre el servidor y el cliente, así sea local, y un explorador Web. El número de clientes concurrentes que pueden conectarse al sistema depende de la capacidad de procesamiento del servidor y de las prestaciones de la red, tanto en el cliente como en el servidor.

- *Infraestructura de comunicación*: comprende los diferentes equipos utilizados para hacer posible la comunicación entre el servidor y los clientes. Los enrutadores, switches, repetidores, proxys y demás equipos que están involucrados en el funcionamiento de la Web son los necesarios para mantener dicha comunicación.

Capítulo 5

Evaluación

En este capítulo se presentan los experimentos realizados para probar la estrategia de producción de oraciones planteada en este trabajo a través de *DISGEN*. En primer lugar, se muestra el esquema de párrafo para introducciones de artículos técnicos tomado como plantilla para construir las unidades del *corpus* y las fuentes de texto usadas para sus contenidos. Las características del *corpus* de texto creado, los escenarios para las pruebas y los criterios de evaluación de las salidas de texto se describen en la siguiente sección. Luego se realiza un resumen de los resultados obtenidos en las salidas del generador de textos, finalmente se presentan las conclusiones del proyecto y las recomendaciones para futuros trabajos.

5.1. El esquema de párrafo empleado y fuentes de contenido

El esquema empleado como plantilla para el párrafo de introducción en artículos técnicos embebido en cada título del *corpus* está formado por seis oraciones. Cada una de estas oraciones tiene un significado especial y se inspira en el estilo de escritura observado en varios escritos de autores para el tema de diseño de amplificadores operacionales en ingeniería electrónica. Cada frase dentro de la oración tiene los siguientes sentidos:

- *Primera oración*: esta oración presenta la idea principal del tema expuesto por el autor a lo largo del artículo. Acá se emplean usualmente términos relacionados con la jerga del tema de estudio, para nuestro caso el diseño de OPAMPs.
- *Segunda oración*: en esta frase el autor hace un complemento o un refuerzo de lo expuesto en la primera oración, es decir, de la idea principal.
- *Tercera oración*: el autor hace la presentación explícita de la primera sección del artículo.
- *Cuarta oración*: el autor presenta la segunda sección del artículo.
- *Quinta oración*: el autor hace la presentación explícita de la tercera sección del artículo.
- *Sexta oración*: esta última frase es empleada para mostrar lo expuesto en las secciones finales del documento, generalmente relacionado con las conclusiones, pruebas, resultados y trabajos futuros.

Se deben considerar las fuentes bibliográficas del contenido de los párrafos. El tema está relacionado con el diseño de amplificadores operacionales en microelectrónica y se toman los artículos respectivos en los catálogos del IEEE e ISCAS. Esta selección la realizan miembros

Tabla 5.1: *Información de la longitud para oraciones.*

Tipo de oración	Longitud máxima	Longitud mínima	Sinónimos indicados
Primera oración	35	9	35
Segunda oración	28	10	30
Tercera oración	22	12	16
Cuarta oración	24	11	14
Quinta oración	22	12	11
Sexta oración	20	9	11

Tabla 5.2: *Información de las partes de voz.*

Parte de voz	RB	DT	IN	NN	NNP	NNS	VBN	VBP	VBZ	VB
Número de apariciones	9	60	54	143	2	29	28	14	18	8

del grupo CIDIC, debido al dominio que poseen sobre ésta área y forman así una colección de textos. En términos de *DISGEN*, estos usuarios cumplen el rol de expertos. A partir de este conjunto, los expertos extraen la información necesaria y construyen las frases siguiendo el formato de la plantilla propuesta para párrafos de introducción. Esta tarea está fuertemente ligada a la subjetividad de las personas encargadas las cuales realizan un consenso para determinar los contenidos, de modo que estén muy relacionados y tener una colección de títulos con un contexto reducido y por lo tanto más similares semánticamente.

Se seleccionan varias personas con el rol de revisores, encargados de supervisar, corregir y aprobar las anotaciones lingüísticas realizadas por los expertos sobre los textos. Se reitera la subjetividad de los revisores que desempeñan esta labor, al influir con su trabajo en la calidad del *corpus* desarrollado, pero se consideran personas cuyo objetivo es refinar las anotaciones lingüísticas. El resultado del ejercicio de revisión define explícitamente el conjunto de títulos empleados finalmente en la validación de la estrategia de producción de oraciones.

Las tablas 5.1 y 5.2 presenta un resumen de las características de la colección de textos en el *corpus*. La oración denominada idea principal en promedio es las más extensas, es decir, se necesitan más palabras para esta clase de oración, esto puede sugerir, la necesidad de más conceptos e ideas para expresar su significado, y si existieran pocos sinónimos, se diría que es más compleja para el proceso de generación. Los resultados respecto a las etiquetas de partes de voz deben ser observados cuidadosamente, pues aunque existe más ocurrencia de sustantivos, los determinantes pueden llegar a ser representativos y para un lingüísta es más conveniente considerar los verbos, adjetivos y sustantivos en caso que estos datos formaran parte de algún estudio gramatical. Éstas consideraciones las realizaron los autores de este proyecto, no obstante, lo adecuado es someter los resultados a personas idóneas en el estudio de la lengua.

5.2. Especificaciones de las pruebas

A partir de la colección textual contruida, se realizan ensayos consistentes en tomar aleatoriamente varios subconjuntos de párrafos con el 10, 20, 50 y 100 por ciento de los títulos del *corpus*. El usuario *editor*, escoge determinada frase sobre la cual desea obtener información, oración objetivo, el sistema forma grupos de frases según el tipo de dicha oración y el porcen-

Tabla 5.3: *A RAIL-TO-RAIL 1-VOLT CMOS OPAMP.*

Tipo de oración	**Texto**
Oración 1	**In this paper, a article only full-CMOS opamp is presented.**
Oración 2	A complementary PMOS-bulk-driven input stage is used to achieve rail-to-rail operation
Oración 3	In the section II describe input-stage
Oración 4	In the section III describe low-voltage CMOS current mirror.
Oración 5	In the section IV shows class AB output stage respective.
Oración 6	Then in section V the complete opamp circuit is ilustrated and HSPICE simultion results are reported.

Tabla 5.4: *Resultados para los subconjuntos del corpus.*

	Estrategia uno			**Estrategia dos**		
Porcentaje	**Puntaje**	**Aciertos**	**Similitud**	**Puntaje**	**Aciertos**	**Similitud**
10	0.6	10	0.4767	0.4	4	0.2945
20	0.6	10	0.5	0.3	5	0.4133
50	0.5	10	0.3931	0.4	5	0.4761
100	0.3	10	0.3839	0.2	7	0.3214

taje del *corpus* especificado, formando así el conjunto de entrada. A partir de éste, el sistema aplica la estrategia de generación indicada por el humano y produce 10 nuevas oraciones, cantidad establecida por ensayo y error, teniendo en cuenta la capacidad del servidor.

El criterio de puntuación del humano, a pesar de ser subjetivo, tiene en cuenta los siguientes factores: que la oración esté bien formada, exprese un significado, se relacione con la oración objetivo y le sirve para la redacción de sus propios textos. No puede esperarse que exista coherencia en las oraciones generadas dado que, como se ha mencionado a través del presente documento, no se abordó el problema a la resolución de dicho aspecto por razones de complejidad.

5.3. Resultados

La oración objetivo para realizar las pruebas fue la primera de la tabla 5.3. En general, este tipo de oraciones presenta mayor variedad de estilos en su escritura respecto a las almacenadas dentro del *corpus*. Esta propiedad facilita la comparación semántica entre la salida producida y las frases de entrada en las estrategias de generación, obteníendose un mayor número de nuevas oraciones relacionadas con el mensaje transmitido en los textos originales.

La tabla 5.4 comprende los valores obtenidos de las pruebas realizadas. Cada fila indica información relativa a la evaluación para las estrategia de generación considerando la puntuación media asignada por los editores a través de la interfaz de *DISGEN*, el número de veces en los cuales la herramienta muestra un valor de similitud y el promedio para este cálculo. Para la primera estrategia, el número de aciertos evidencia que dentro de la población siempre existió relación semántica con la oración objetivo, además, a medida que se emplean más entradas, disminuye esta similitud, lo cuál se confirma con la valoración de los editores. Para el caso de la segunda estrategia, no se puede determinar una tendencia respecto al valor de similitud calculado y el puntaje del editor, pero, a medida que aumenta el número de oracio-

nes de entrada existe mayor posibilidad de encontrar similitud, mostrando la efectividad de la estrategia en cuánto a generar frases que contengan términos usados en los textos originales.

Las características de los textos de entrada empleados para las pruebas fueron determinantes en los resultados producidos por los procedimientos de generación. El tamaño del *corpus* construído dependió en gran medida del trabajo realizado por los miembros del CIDIC, quienes adaptaron los textos de artículos técnicos a la plantilla para párrafo y se encargaron de la anotación lingüística con el apoyo de la herramienta *DISGEN*. Lo anterior implica tiempo que puede llegar a ser considerable si se tiene en cuenta la habilidad de los expertos, su disposición para desempeñar estas labores y su adecuación al prototipo de software. De esta manera, sería conveniente realizar una segunda etapa involucrando mayor cantidad de expertos en la tarea de agregar títulos al *corpus*, con considerables anotaciones lingüísticas y diferentes plantillas de párrafo para comprobar de forma más precisa la validez de las estrategias de generación.

5.4. Conclusiones y observaciones

- Teniendo en cuenta la carencia de metodologías para medir la eficiencia de los sistemas de GLN, resultó interesante plantear la similitud semántica como criterio de evaluación. La dificultad radica en la ausencia de expresiones matemáticas que permitan modelar de forma completa el lenguaje natural y con base en ellas determinar funciones o métodos de optimización. Por esta razón no se aplicaron técnicas propias de la inteligencia artificial, como redes neuronales, lógica borrosa, algoritmos genéticos, entre otras, las cuales son válidas en la resolución de problemas representados a través de modelos analíticos y numéricos explícitos. No obstante, puede considerarse su aplicación en tareas intermedias del proceso de generaci ón, tal como se hace uso de la teoría de las máquinas de soporte vectorial en el etiquetado de partes de voz del *SVMTagger*. Estas premisas deben tomarse como referencia para trabajos futuros, los cuales pueden representar un aporte importante al estudio en el área.

- El trabajo descrito en este proyecto representa una primera aproximación a la generación de lenguaje natural en el ámbito local de la Universidad Industrial de Santander. El trabajo es significativo si se tiene en cuenta la complejidad del problema, los experimentos realizados y los procedimientos diseñados tendientes a abordar un fenómeno social, como lo es el lenguaje natural, de forma técnica, aprovechando el conocimiento de los trabajos existentes en la literatura.

- El procedimiento de elaboración del *corpus* permitió establecer una guía para la organización de información textual aplicable a computación lingüística. Lo anterior puede ser empleado por expertos, no sólo para realizar estudios de comportamiento lingüístico en inglés, sino para apoyar el aprendizaje y enriquecimiento de competencias gramaticales en personas que desean adquirir dominio en la redacción de textos.

- A partir de los experimentos llevados a cabo en el primer prototipo donde se adaptaron las herramientas *Dada engine*, *SVMTagger*, *Sense Relate*, *Wordnet* y *Moby Thesaurus* bajo un programa que realizaba textos por sustitución de palabras, se determinó la dificultad de dejar al computador este proceso de manera totalmente automática. Esto se debe a las limitaciones de cada una de ellas al momento de emplearlas en GLN para el contexto de diseño de circuitos integrados y a la falta de herramientas de código abierto más completas. Sin embargo, estas herramientas sirvieron de apoyo en labores intermedias que involucran la intervención de expertos en el tema.

- A partir de una documentación bibliográfica exhaustiva, el primer capítulo refleja que el área de la GLN no sólo se enfoca de aspectos teóricos, también a aplicaciones de gran escala para los cuales ofrece ayuda parcial o total a la resolución de determinados problemas. Lo anterior permitió establecer como objetivo la generación de oraciones en inglés, considerando la necesidad de apoyar la redacción de escritos y la gran cantidad de desarrollos reportados para el mencionado idioma.

- La ayuda humana en labores intermedias de generación fue clave al momento de diseñar la fuente de conocimiento, especialmente en la construcción del corpus y en la evaluación de los resultados. Es posible realizar un corpus lingüístico con el apoyo de expertos y empleando recursos informáticos disponibles en el área de la computación lingüística. Por otro lado, la evaluación subjetiva es comparada con el criterio de similitud semántica buscando obtener más argumentos para analizar los resultados de la salida del generador.

- Los fundamentos de las estrategias de producción de frases permiten ser asociadas con teorías aplicables a la GLN. La primera estrategia podría pensarse como un procedimiento de búsqueda clásico en inteligencia artificial debido a la forma de originar el árbol, donde las ramas representan el conjunto de términos con los cuales se crea la frase. La segunda, por su parte, se relaciona con la teoría de autómatas estocásticos al momento de generar las secuencias de etiquetas gramaticales por el uso de probabilidades. Para los dos casos, estas consideraciones deben tomarse cuidadosamente y revisar a profundidad la correspondiente fundamentación teórica con el fin de perfeccionar las técnicas y mejorar los resultados.

- El desarrollo del sistema *DISGEN* evidencia que es posible construir aplicaciones enfocadas al apoyo de procesos en campos de desempeño alternativos y novedosos. Aunque el trabajo en este sentido dió como resultado un prototipo de software, se pudo observar la viabilidad de emplear modelos de desarrollo satisfactoriamente, análogo al desempeño mostrado en el contexto de las aplicaciones comerciales. La representación con casos de uso fue la adecuada para modelar el rol de los actores en el sistema, pudiendo determinar el grado de relevancia de la información presentada por los textos producidos para cada usuario.

- La adaptación de herramientas necesarias en el proceso de generación, motivó el uso del lenguaje de programación *PHP*, el manejador de bases de datos *MySQL*, el etiquetador gramatical *SVMTagger* y los diccionarios léxicos *WordNet* y *Moby Thesaurus* que lograron adecuarse en entorno Web, salvo por el tiempo computacional en la ejecución del etiquetador. El modelo de entidad-relación para la base de datos soportada por *DISGEN* almacenó la información del *corpus* y los elementos relacionados con la operación del sistema conforme a los requerimientos. En este sentido, se logró seleccionar y aplicar procesos de adaptacion de software libre a una herramienta computacional relacionada con generación de lenguaje natural.

- En conclusión, el trabajo cumple con los objetivos trazados originalmente y contribuye en la presentación de un nuevo área de trabajo dentro de la Universidad Industrial de Santander. No obstante, se reitera el carácter de aproximación en el tema de la GLN que particularmente se abordó durante el presente documento. Se espera puedan derivarse proyectos relacionados que aporten en la construcción de un *corpus* representativo en el área de ingeniería electrónica, perfeccionen las técnicas empleadas en las estrategias de generación y los criterios de evaluación de las salidas de texto.

5.5. Recomendaciones para trabajos futuros

- La imposibilidad de abordar el problema de la coherencia textual usando la *RST* tal como se consideró en la revisión reportada en el capítulo 3 y la necesidad de este tipo de aplicaciones dirigen el interés hacia el desarrollo de herramientas automatizadas para el análisis retórico de texto. Una premisa importante a tener en cuenta para emprender esta labor, es la realización de un estudio respecto a las formas de escritura según el estilo de los autores en determinadas área de conocimiento, involucrando expertos en lingüística.

- Debido a la importancia de poseer un *corpus* anotado con información de relaciones de discurso y ante la dificultad expresada por los autores al no contar con la capacitación necesarias para realizarlo, se plantea el desarrollo de proyectos, en los cuales, inicialmente se realice el estudio correspondiente con lingüístas sobre un tema específico, luego se codifiquen sus resultados y se empleen posteriormente como recurso computacional para el enriquecimiento del *corpus*.

- Motivados por el problema tratado por la herramienta *SVMTagger* y considerando algunas falencias en su diseño, en especial el tiempo computacional, se propone la elaboración de un etiquetador para marcas gramaticales, basado en el *corpus* planteado en este trabajo. Dada la variedad de métodos que se han desarrollado para esta tarea en procesamiento de lenguaje natural, se recomienda abordarlo empleando técnicas de razonamiento aproximado, tales como redes neuronales, lógica difusa a incluso algoritmos genéticos.

- Se ha reiterado en diferentes oportunidades la necesidad de personas expertas en el estudio del lenguaje natural, desde el punto de vista propio de la teoría lingüística. A partir de esta apreciación y las ventajas de disponer de un *corpus* textual, resulta viable la construcción de un diccionario de definiciones léxico-semánticas para términos en ingeniería electrónica. Además, pueden llevarse a cabo estudios respecto al estilo de escritura adoptado por los ingenieros eletrónicos de la UIS en textos de artículos técnicos en inglés y otro relativo al impacto de la herramienta *DISGEN* en el aprendizaje de redacción de textos en inglés. Bajo estas circunstancias, lo más importante es considerar en el marco de trabajo a personas expertas en el estudio del idioma, como pueden ser miembros de la Escuela de Idiomas y el Instituto de Lenguas de la Universidad.

- El desarrollo de un marco metodológico mejorado para la generación de textos, con base en las estrategias planteadas para *DISGEN* puede dar, de manera más específica, continuidad al presente trabajo. Sin embargo, se deben resaltar las limitaciones que plantea el estudio en GLN, especialmente las relacionadas con el idioma y el área de conocimiento sobre el que se desea producir el texto.

- Desde el punto de vista de la ingeniería de software, es pertinente la realización de un análisis de la influencia ejercida por el desarrollo de sistemas de GLN en aplicaciones comerciales en ámbito local. Todo el conocimiento reportado en el tema puede ser aprovechado para diseñar e implementar herramientas relacionadas con la redacción de textos en inglés al ser este considerado el idioma universal.

- Finalmente, la línea de investigación futura puede ir dirigida al desarrollo de metodologías de evaluación de calidad para las salidas de texto en sistemas de GLN. Esto resulta

de particular interés porque el estado del arte en este aspecto lo reporta como poco explorado debido a su alta complejidad, dejándolo únicamente a criterios subjetivos que impiden garantizar completamente la calidad de los textos generados.

Apéndice A

Detalles de los diagramas de casos de uso

A.1. Principales actores de *DISGEN*

A continuación se relacionan los tipos de actores y se hace una corta descripción de su responsabilidad en el sistema.

Actor invitado
Información que le suministra el sistema ▪ Contenido del trabajo realizado en el proyecto. ▪ Referencia para contacto con personas relacionadas con el desarrollo del proyecto. ▪ Texto generado por el generador de oraciones.

Actor administrador
Responsabilidad en el sistema ▪ Gestionar las cuentas de usuarios para experto, editor y revisor. ▪ Crear las especificaciones de los esquemas para párrafos que pueden ser creados dentro de un *corpus* de texto.
Información que le suministra el sistema ▪ Datos básicos de los usuarios tipo experto, revisor y editor.

Actor experto
Responsabilidad en el sistema ▪ Insertar y manejar la información textual de los *corpus*. ▪ Realizar la anotación lingúistica sobre los *corpus* de texto.
Información que le suministra el sistema ▪ Las oraciones del *corpus* que han sido aprobadas por el revisor para el proceso de generación. ▪ Las notas de corrección hechas por el revisor sobre los *corpus* ya anotados lingüísticamente.

Actor revisor
Responsabilidad en el sistema ▪ Realizar la revisión de los *corpus* anotados que han sido insertados por los usuarios tipo experto ▪ Dar sugerencias para las correcciones respectivas en los *corpus* anotados. ▪ Aprobar las oraciones anotadas que se toman como base para el procedimiento de generación.
Información que le suministra el sistema ▪ Información de los *corpus* de texto anotados pertinentes que ayuden a realizar la revisión.

Actor editor
Responsabilidad en el sistema ▪ Evaluar la calidad de los textos generados por el sistema.
Información que le suministra el sistema ▪ Texto producido con el generador de oraciones.

A.2. Diagrama de casos de uso de modelado de requisitos

La figura 4.11 muestra el diagrama de casos de uso del modelado de requisitos para *DISGEN*. Seguido, se especifica el comportamiento de cada caso de uso en forma textual que permiten relacionar las funcionalidades del sistema con los requisitos.

Consulta de información general del proyecto: el sistema debe permitir a los invitados visualizar la información sobre el proyecto, en una interfaz amigable e intuitiva que le permita conocer el marco de estudio de la GLN y las especificaciones generales del trabajo propuesto en el presente proyecto.

Gestión de usuarios: el sistema debe poder gestionar la actualización de las cuentas de los usuarios tipo experto, editor y revisor teniendo en cuenta sus características. La gestión de usuarios es realizada por el usuario tipo administrador.

Ingreso al sistema: el sistema debe validar e identificar el tipo de usuario que se conecta al sistema, ya sea el administrador, los expertos, los revisores y los editores.

Gestión de esquemas para párrafos: el sistema debe permitir la inserción de datos relacionados con los tipos de párrafos que pueden ser creados por los expertos en los *corpus*. Dicha información debe tener en cuenta el número de oraciones y la posición dentro párrafo, junto con las descripciones respectivas para los tipos de párrafos y frases que los componen. Al eliminar y modificar estos esquemas, se requiere validar que no existan *corpus* que los empleen como plantillas para el texto. El usuario encargado de realizar esta gestión es el administrador.

Gestión de *corpus* lingüísticos: el sistema debe permitir el ingreso, modificación y eliminación de los *corpus* lingüísticos de texto, teniendo en cuenta su contenido y los esquemas para párrafos que pueden emplearse como plantilla. Esta gestión se hace especificando el título, los párrafos y el comentario general sobre el texto que puede realizar el usuario. La gestión de los *corpus* lingüísticos la realiza el experto.

Gestión de *corpus* lingüísticos anotados: el sistema debe dar la posibilidad al experto de gestionar los *corpus* anotados de texto, teniendo en cuenta su contenido, los esquemas de párrafo y la tarea de anotación lingüística sobre éstos. A partir de un *corpus* de texto no anotado ya insertado por el experto, debe permitir su respectiva anotación lingüística.

Anotación lingüística: el sistema debe implementar la tarea de anotación lingüística de partes de voz sobre oraciones de textos, tanto con el etiquetado automático como de la corrección manual por parte del humano. Adicionalmente, esta anotación debe permitir incluir los sinónimos para las palabras en las oraciones e incluir notas descriptivas o aclaratorias sobre ésta.

Revisión de *corpus* anotados: el sistema debe proveer una interfaz para realizar la revisión de los *corpus* de texto anotados lingüísticamente. Esto incluye correcciones sobre las anotaciones realizadas y notas aclaratorias sobre la revisión. Además, debe permitir dar aprobación a los *corpus* anotados que se emplean para realizar la generación de frases y restringir la posterior edición lingüística de los textos por parte de los expertos. Esta labor es realizada por el actor revisor.

Estrategia de generación: el sistema debe ejecutar las estrategias de generación planteadas en el presente trabajo. Se deben tener en cuenta los *corpus* que han sido aprobados por el revisor y el algoritmo de similitud entre frases.

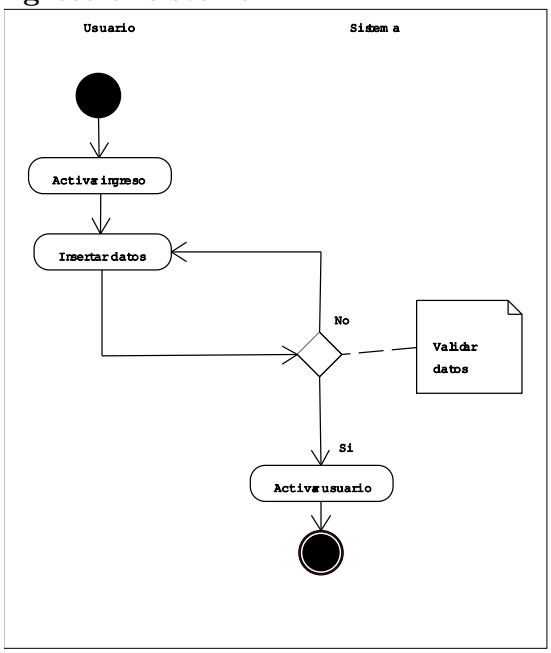
Algoritmo de similitud entre frases: el sistema debe contener la implementación del procedimiento que determina la similitud entre oraciones, para emplearla en los procesos fundamentales de generación de texto.

Generación aleatoria de frases: el sistema debe realizar la producción de oraciones con base en las estrategias de generación planteadas en el trabajo. El invitado puede probar una demostración del generador incluída en la página de presentación del proyecto.

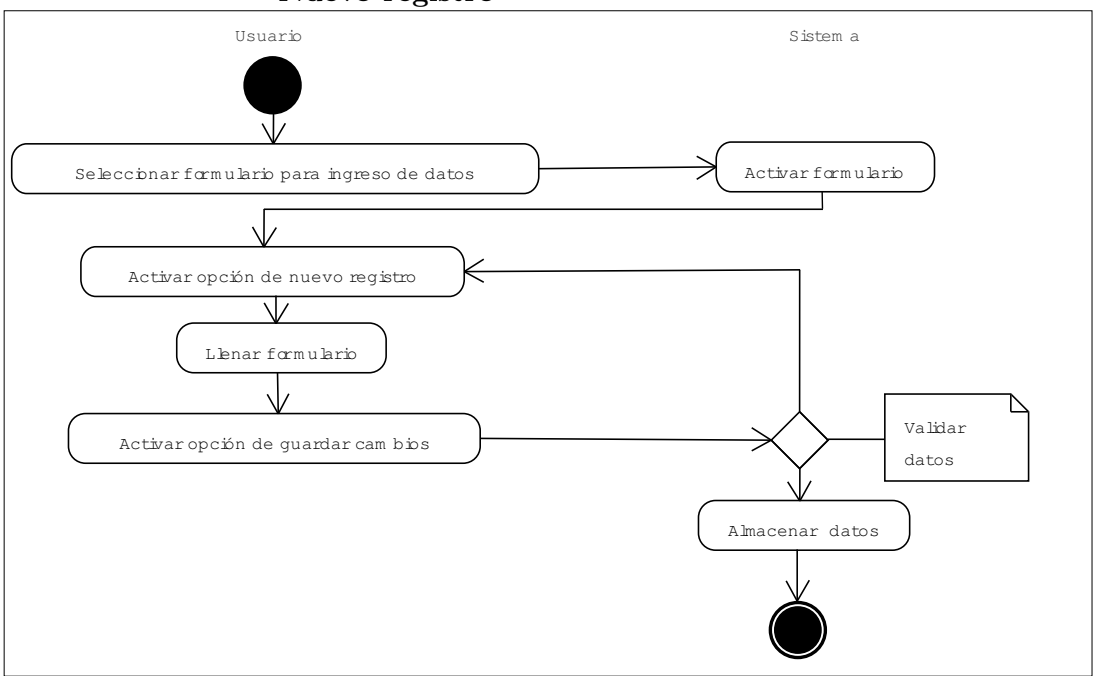
Evaluación del texto generado: el sistema debe almacenar información relacionada con la evaluación hecha por los editores sobre los textos generados por la herramienta. Esta evaluación se realiza indicando una puntuación en escala numérica.

A.3. Diagramas de actividades de las operaciones básicas del sistema

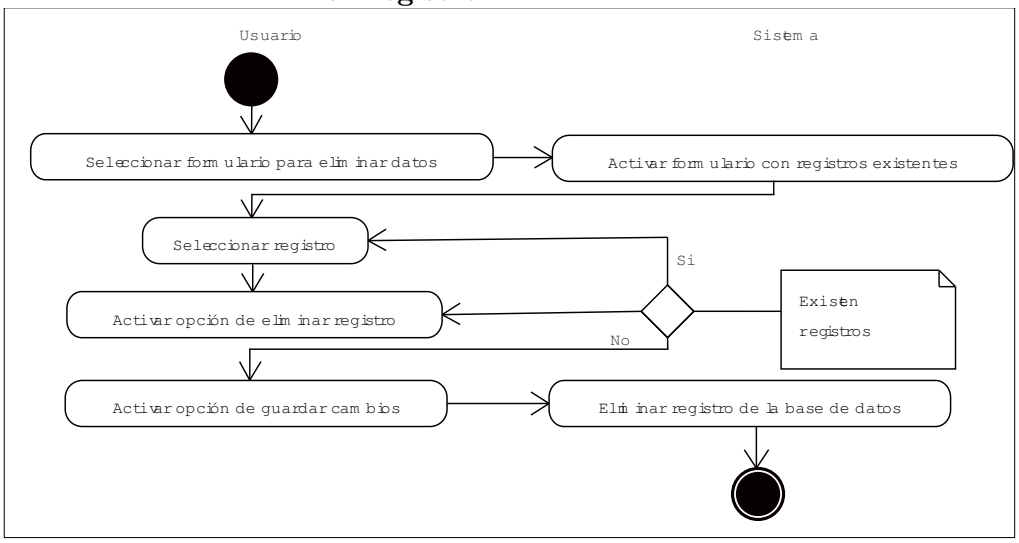
Ingreso al sistema



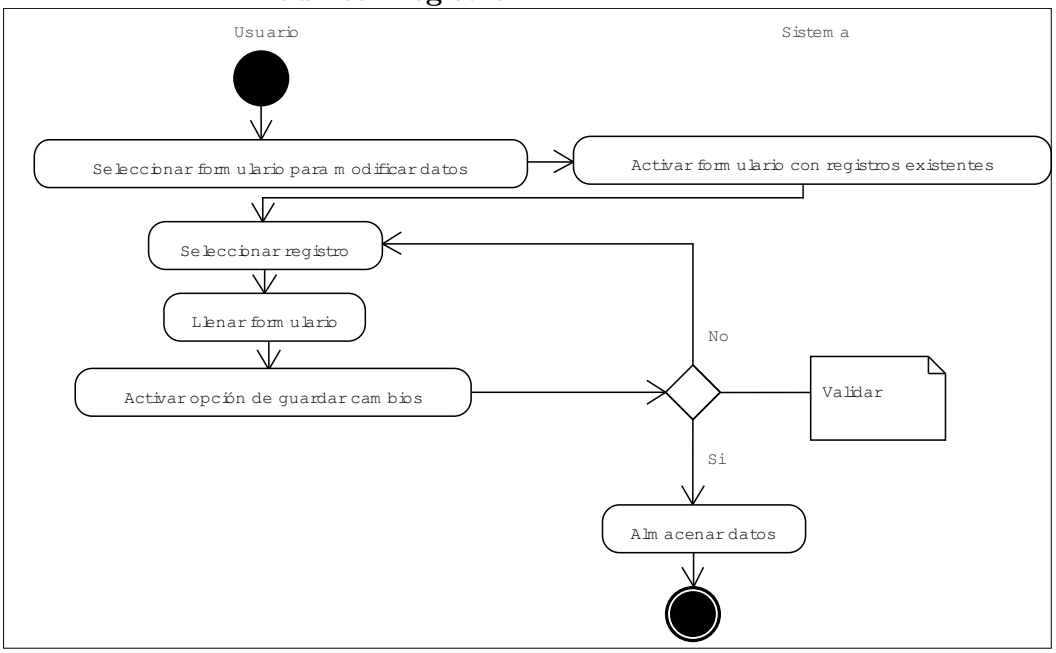
Nuevo registro



Eliminar registro



Modificar registro



Apéndice B

Diccionario de datos

Deben tenerse en cuenta algunas consideraciones. Los campos subrayados corresponden a las llaves primarias de las tablas y los campos acompañados con -> son llaves foráneas cuya referencia respectiva se indica enseguida en la forma *tabla(campo)*. La notación de los tipos de datos es tomada de los tipos soportados por *MySQL*.

Tabla usuario

CAMPO	TIPO	NULO	DESCRIPCIÓN
id_usuario	int(11)	No	Identificador de usuario.
nombre	mediumtext	No	Nombres del usuario.
apellido	mediumtext	No	Apellidos del usuario.
login	mediumtext	No	Login del usuario. Encriptado.
password	mediumtext	No	Password del usuario. Encriptado.
correoe	mediumtext	No	Dirección de correo electrónico del usuario.
tipo_usuario->tipo_usuario(id_tipo_usuario)	smallint(6)	No	Identificador que referencia el tipo de usuario.

Tabla usuario_conectado

CAMPO	TIPO	NULO	DESCRIPCIÓN
usuario->usuario(id_usuario)	int(11)	No	Identificador que referencia el tipo de usuario.

Tabla tipo_usuario

CAMPO	TIPO	NULO	DESCRIPCIÓN
id_tipo_usuario	smallint(6)	No	Identificador del tipo de usuario. Tiene los siguientes valores: 1 para administrador, 2 para editor, 3 para experto y 4 para revisor.
tipo_usuario	mediumtext	No	Nombre del tipo de usuario.
descripcion	mediumtext	Si	Descripción relacionada con el tipo de usuario.

Tabla tipo_parrafo

CAMPO	TIPO	NULO	DESCRIPCIÓN
id_tipo_parrafo	int(11)	No	Identificador del tipo (esquema) de párrafo.
tipo_parrafo	mediumtext	No	Nombre del tipo de párrafo.
descripcion	mediumtext	No	Descripción relacionada con el tipo de párrafo.

Tabla tipo_oracion

CAMPO	TIPO	NULO	DESCRIPCIÓN
id_tipo_oracion	int(11)	No	Identificador del tipo (esquema) de oración.
tipo_oracion	mediumtext	No	Nombre del tipo de oración.
descripcion	mediumtext	No	Descripción relacionada con el tipo de oración.
posicion	smallint(6)	No	Posición de la oración dentro de la oración. Por ejemplo, el valor de 1 es para la primera oración, 2 para la segunda oración y así sucesivamente.
tipo_parrafo->tipo_parrafo(id_tipo_parrafo)	int(11)	No	Referencia al tipo de párrafo al que pertenece la oración.

Tabla etiqueta_parte_de_voz

CAMPO	TIPO	NULO	DESCRIPCIÓN
id_etiqueta	int(11)	No	Identificador de la etiqueta de parte de voz.
etiqueta	mediumtext	No	Notación de la etiqueta de parte de voz. Casi todos los valores de esta tabla corresponden con el conjunto de etiquetas de partes de voz del *Penn TreeBank*.
descripcion	mediumtext	No	Descripción del significado de la parte de voz.

Tabla titulo

CAMPO	TIPO	NULO	DESCRIPCIÓN
id_titulo	bigint(20)	No	Identificador del título (unidad de *corpus*).
titulo	mediumtext	No	Título de la unidad de *corpus*.
propietario->usuario(id_usuario)	int(11)	No	Referencia al usuario tipo experto que gestiona el título.
nota	mediumtext	Si	Descripción relacionada con el título.

Tabla oracion

CAMPO	TIPO	NULO	DESCRIPCIÓN
id_oracion	bigint(20)	No	Identificador de la oración.
titulo->titulo(id_titulo)	bigint(20)	No	Referencia al título al que pertenece la oración.
tipo_oracion->tipo_oracion(id_tipo_oracion)	int(11)	No	Referencia al tipo de la oración.

Tabla termino

CAMPO	TIPO	NULO	DESCRIPCIÓN
oracion->oracion(id_oracion)	bigint(20)	No	Identificador de la oración a la que pertenece el término.
posicion	smallint(6)	No	Posición del término dentro de la oración. Por ejemplo el valor de 1 es para el promer término, 2 para el segundo término y así sucesivamente.
palabra->palabra(id_palabra)	bigint(20)	No	Referencia a la palabra que constituye en término.
forma	mediumtext	No	Notación de la capitalización del término. Ver la descripción de dicha notación en la siguiente sección.

Tabla palabra

CAMPO	TIPO	NULO	DESCRIPCIÓN
id_palabra	bigint(20)	No	Identificador de la palabra.
palabra	mediumtext	No	Unidad básica de texto (en minúscula).

Tabla parte_de_voz

CAMPO	TIPO	NULO	DESCRIPCIÓN
oracion->oracion(id_oracion)	bigint(20)	No	Referencia a la oración que tiene la parte de voz.
posicion	smallint(16)	No	Posición dentro de la oración dónde se encuentra la parte de voz. 1 para la primera, 2 para la segunda y así sucesivamente.
parte_de_voz->etiqueta_parte_de_voz(id_etiqueta)	int(11)	No	Referencia a la etiqueta de parte de voz asignada.

Tabla sinonimo

CAMPO	TIPO	NULO	DESCRIPCIÓN
oracion->oracion(id_oracion)	bigint(20)	No	Referencia a la oración a la que pertenece el sinónimo.
palabra->palabra(id_palabra)	bigint(20)	No	Referencia a la palabra que indica el sinónimo.
posicion	smallint(6)	No	Posición dentro de la oración dónde se encuentra el sinónimo. 1 para la primera, 2 para la segunda y así sucesivamente.
forma	mediumtext	No	Notación de la capitalización del sinónimo. Ver la descripción de dicha notación en la siguiente sección.

Tabla oracion_etiquetada

CAMPO	TIPO	NULO	DESCRIPCIÓN
oracion->oracion(id_oracion)	bigint(20)	No	Referencia a la oración que ha sido anotada lingüísticamente.
nota	mediumtext	Si	Descripción relacionada con la oración anotada.

Tabla oracion_revisada

CAMPO	TIPO	NULO	DESCRIPCIÓN
oracion->oracion(id_oracion)	bigint(20)	No	Referencia a la oración anotada que ha sido revisada.
revisor->usuario(id_usuario)	int(11)	No	Referencia al usuario tipo revisor que supervisa la oración anotada.
nota	mediumtext	Si	Descripción relacionada con la revisión de la oración anotada.
aprobada	tinyint(1)	No	Banderilla que indica si la oración anotada puede utilizarse para el proceso de generación de nuevas oraciones. Sus valores son true o false.
bloqueada	tinyint(1)	No	Banderilla que indica si la oración anotada puede ser editada por el experto que hizo su inserción. Sus valores son true o false.

Tabla similitud

CAMPO	TIPO	NULO	DESCRIPCIÓN
oracion1->oracion(id_oracion)	bigint(20)	No	Referencia a la primera oración anotada.
oracion2->oracion(id_oracion)	bigint(20)	No	Referencia a la segunda oración anotada.
similitud	double	No	Valor numérico de la similitud entre las oraciones referenciadas en los campos oracion1 y oracion2, calculado con el algoritmo de similaridad propuesto en el presente trabajo.

Tabla oracion_evaluada

CAMPO	TIPO	NULO	DESCRIPCIÓN
original->oracion(id_oracion)	bigint(20)	No	Referencia a la oración anotada tomada como oración objetivo en el proceso de generación.
generada	mediumtext	No	Texto con la oración generada por el sistema.
puntuacion	double	No	Valor numérico del puntaje asignado por los editores a la oración generada con respecto a la calidad.

Tabla parametros

CAMPO	TIPO	NULO	DESCRIPCIÓN
L	mediumtext	Si	Opción del SVMTagger*.
I	mediumtext	Si	Opción del SVMTagger*.
V	mediumtext	Si	Opción del SVMTagger*.
S	mediumtext	Si	Opción del SVMTagger*
T	mediumtext	Si	Opción del SVMTagger*
B	mediumtext	Si	Opción del SVMTagger*
K	mediumtext	Si	Opción del SVMTagger*
U	mediumtext	Si	Opción del SVMTagger*
SVMTagger	mediumtext	Si	Ruta del SVMTagger en el servidor.
WSJTP	mediumtext	Si	Localización del modelo entrenado del *corpus* que emplea el SVMTagger para realizar la anotación automática.
result_por_pagina	mediumtext	Si	Número de resultados por página que aparecerán en las diferentes pantallas de los módulos.
tiempo_bien	mediumtext	Si	Tiempo en milisegundos que tardará la página de respuesta al realizar operaciones ejecutadas satisfactoriamente.
tiempo_error	mediumtext	Si	Tiempo en milisegundos que tardará la página de respuesta al realizar operaciones que no puedan terminar satisfactoriamente.
simbolos	mediumtext	Si	Símbolos textuales que se tienen en cuenta para realizar la separación de términos en los textos.
simbolos_espacio_antes	mediumtext	Si	Símbolos textuales tenidos en cuenta para formar las frases, con un espacio en blanco antes de su aparición.
simbolos_espacio_despues	mediumtext	Si	Símbolos textuales tenidos en cuenta para formar las frases, con un espacio en blanco después de su aparición.
dir_temp	mediumtext	Si	Ruta del directorio en el servidor dónde se crean los archivos temporales de la anotación lingüística.

(*) Consulte la documentación del SVMTool. http://www.lsi.upc.edu/ñlp/SVMTool/

Apéndice C

Vista de implementación

El sistema DISGEN está implementado en archivos de texto plano, con codificación UTF-8. El tamaño total de la aplicación es de 3.8 MB, y los archivos se encuentran almacenados en diferentes carpetas ubicadas en el directorio */var/www* del servidor. Estos archivos se dividen principalmente en los siguientes tipos:

Archivos de PHP: son archivos con extensión *.php* escritos en lenguaje PHP que contienen los métodos de respuesta a las solicitudes de los clientes.

Archivos de hojas de estilo: son archivos con extensión *.css* que contienen la definición de las hojas de estilo en cascada usadas por el sistema para la presentación de las interfaces y datos a los usuarios.

Archivos para volcado de base de datos: estos archivos, con extensión *.sql*, contienen instrucciones en *SQL* para la instalación de la base de datos inicial del sistema, o llegado el caso restaurar el contenido de la base de datos actual.

Archivos de comando JavaScript: son archivos con extensión *.js*, escritos en forma procedimental, que contienen secuencias de comandos en JavaScript para dar respuesta a los eventos del usuario, realizar validaciones en línea y algunos efectos dinámicos en la interfaz.

Imágenes: contienen gráficos para la presentación de las interfaces de usuario. Están en diversos formatos, como *.gif*, *.png*, *.bmp* y otros posibles.

Las carpetas que conforman el sistema son las siguientes:

disgen: es la carpeta principal dónde se instala el sistema. Contiene el archivo *index.html* que es el punto de partida de la aplicación y se redirecciona hacia el módulo del invitado.

disgen.administrator: esta subcarpeta contiene archivos utlizados para el funcionamiento del módulo de administrador.

disgen.bd: contiene el archivo de volcado de la base de datos inicial del sistema.

disgen.css: se guarda la hojas de estilo en cascada del sistema, empleada para el formato y estilo de las interfaces de usuario.

disgen.editor: esta subcarpeta contiene archivos utlizados para el funcionamiento del módulo de editor.

disgen.expert: esta subcarpeta contiene archivos utlizados para el funcionamiento del módulo de experto.

disgen.img: aquí se guardan las imágenes utilizadas para construir el entorno gráfico de usuario, tales como iconos, logos, botonos de menúes, entre otras.

disgen.invited: esta subcarpeta contiene archivos utlizados para el funcionamiento del módulo de invitado.

disgen.js: se guardan las secuencias de comandos en Javascript requeridas por los demás módulos.

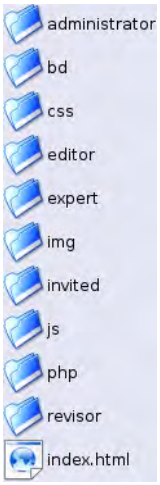


Figura C.1: *Organización de las carpetas de DISGEN.*

disgen.php: aquí se guardan los archivos escritos en PHP empleados en los demás módulos y los archivos de parámetros para la configuración general del sistema. Contiene el archivo *insertadmin.php* que permite ingresar la cuenta del administrador a la base de datos del sistema, por lo que se recomienda cambiarlo de ubicacion o suprimirlo una vez montada la aplicacion.

disgen.revisor: esta subcarpeta contiene archivos utlizados para el funcionamiento del módulo de revisor.

Se muestra en la figura C.1 la organización de las carpetas en DISGEN.

Apéndice D

Instalación y configuración

D.1. Requerimientos de software

Se requiere del siguiente componente software y configuración en el cliente:

- Un navegador Web, preferiblemente *Mozilla Firefox 1.5.0.5.* con soporte habilitado para ejecutar secuencias de comandos de Javascript embebidas en páginas Web.

El servidor debe tener estos componentes instalados:

- Sistema operativo con prestaciones de servidor, se recomienda *GNU/Linux.* Se usó para las pruebas *Linux Kubuntu 6.06* en un computador portátil y *Linux Debian Sarge 3.1* en una máquina servidor.
- Servidor Web con soporte para *PHP 4.3.10* o superior. Se recomienda el uso de *Apache 1.3.33* o superior.
- Lenguaje de programación *PHP 4.3.10* o superior con soporte habilitado para sesiones..
- Manejador de base de datos *MySQL 4.0.24* o superior.
- Diccionario léxico *WordNet 2.0.*
- Servidor de base de datos léxica *Dict* con diccionario *Moby thesaurus* instalado.
- Etiquetador gramatical *SVMTool 1.1.4* en versión para *C++.*

D.2. Requerimientos de hardware

Para el cliente:

- Resolución de pantalla de 800*600 (mínimo) o superior de 1024*768 (recomendado) a color verdadero.
- Conexión a Internet, o forma de comunicación a través del protocolo HTTP sea localmente o a través de una Intranet.

Fue usado como servidor de prueba un equipo con las siguientes especificaciones:

- Conexión a Internet de velocidad media.
- Procesador AMD SEMPRON de 1.8 Ghz.
- Memoria RAM de 256 Megabytes.
- Capacidad de disco de 40 Gigabytes.

Sin embargo, se estima, dependiendo del número usuarios concurrentes, un servidor con las siguientes prestaciones:

- Conexión a Internet de alta velocidad.
- Procesador de cualquier tecnología a 3.2 Ghz o superior.
- Memoria RAM de 1 Gygabyte o superior.
- Capacidad en disco de 160 Gigabytes o más.

D.3. Procedimiento de instalación

1. **Comprobación de los requerimientos de instalación**: para ello se inspecciona el software requerido, es decir, el sistema operativo *GNU/Linux*, el servidor Web *Apache*, el lenguaje *PHP* con soporte habilitado para sesiones y el manejador de bases de datos *MySQL*.

   Es importante tener en cuenta la ruta de instalación de herramienta *SVMTagger*, por lo general ubicada en la carpeta *bin* del *SVMTool*, debido a que debe ser especificada en la tabla de parámetros dentro de la base de datos. Si se tiene instalado el *SVMTagger* en la ruta */usr/local/SVMTool114/bin*, es posible realizar la comprobación de la herramienta escribiendo en un intérprete de comandos la sentencia

   ```
   /usr/local/SVMTool114/bin/SVMTagger
   ```

   la cuál desplegaría la salida mostrada en la figura D.1.

   Compruebe que *Wordnet* esté instalado, digitando en una consola de Linux el comando

   ```
   wn
   ```

   la cuál desplegará un resultado similar al presentado en la figura D.2.

   Así mismo, digitando en el intérprete de comandos

   ```
   dict -D
   ```

   puede obtenerse la salida de la figura D.3.

   Para información relacionada con esta comprobación, se recomienda consultar la documentación respectiva.

```
SVMTool++ v 1.1.2 -- SVMTagger

Usage : svmt [options] <model>  < stdin > stdout

options:
        -L or -l <Window Lenght>
                <Window Lenght> have to be greater than 2
                7 (default)
        -I or -i <Interest Point>
                <Interest Point> have to be greater than 0
                4 (default)
        -S or -s <direction>
                LR      left-to-right   (default)
                RL      right-to-left
                LRL     both left-to-right and right-to-left
        -T or -t <strategy>
                0       one-pass   [default - requires model 0]
                1       two-passes [revisiting results and relabeling - requires model 2 and model 1]
                2       one-pass   [robust against unknown words - requires model 0 and model 2]
                3       one-pass   [unsupervised learning models - requires model 3]
                4       one-pass   [very robust against unknown words - requires model 4]
                5       one-pass   [sentence-level likelihood - requires model 0] Not implemented!!
                6       one-pass   [robust sentence-level likelihood - requires model 4] Not implemented!!
        -B or -b <backup_lexicon>
        -K <n> weight filtering threshold for known words (default is 0)
        -U <n> weight filtering threshold for unknown words (default is 0)
        -V or -v verbose

model: model location (path/name)

Usage : SVMTagger -V -S LRL -T 0 /home/users/me/SVMTool/models/eng/WSJTP < WSJTP.TEST > WSJTP.TEST.OUT
```

Figura D.1: *Vista de consola para SVMTagger.*

```
usage: wn word [-hgla] [-n#] -searchtype [-searchtype...]
       wn [-l]

        -h              Display help text before search output
        -g              Display gloss
        -l              Display license and copyright notice
        -a              Display lexicographer file information
        -o              Display synset offset
        -s              Display sense numbers in synsets
        -n#             Search only sense number #

searchtype is at least one of the following:
        -ants{n|v|a|r}          Antonyms
        -hype{n|v}              Hypernyms
        -hypo{n|v}, -tree{n|v}  Hyponyms & Hyponym Tree
        -entav                  Verb Entailment
        -syns{n|v|a|r}          Synonyms (ordered by estimated frequency)
        -smemn                  Member of Holonyms
        -ssubn                  Substance of Holonyms
        -sprtn                  Part of Holonyms
        -membn                  Has Member Meronyms
        -subsn                  Has Substance Meronyms
        -partn                  Has Part Meronyms
        -meron                  All Meronyms
        -holon                  All Holonyms
        -causv                  Cause to
        -pert{a|r}              Pertainyms
        -attr{n|a}              Attributes
        -deri{n|v}              Derived Forms
        -domn{n|v|a|r}          Domain
        -domt{n|v|a|r}          Domain Terms
        -faml{n|v|a|r}          Familiarity & Polysemy Count
        -framv                  Verb Frames
        -coor{n|v}              Coordinate Terms (sisters)
        -simsv                  Synonyms (grouped by similarity of meaning)
        -hmern                  Hierarchical Meronyms
        -hholn                  Hierarchical Holonyms
        -grep{n|v|a|r}          List of Compound Words
        -over                   Overview of Senses
```

Figura D.2: *Vista de consola para WordNet.*

```
Databases available:
 gcide          The Collaborative International Dictionary of English v.0.48
 vera           V.E.R.A. -- Virtual Entity of Relevant Acronyms (March 2005)
 moby-thesaurus Moby Thesaurus II by Grady Ward, 1.0
```

Figura D.3: *Vista de consola para Dict.*

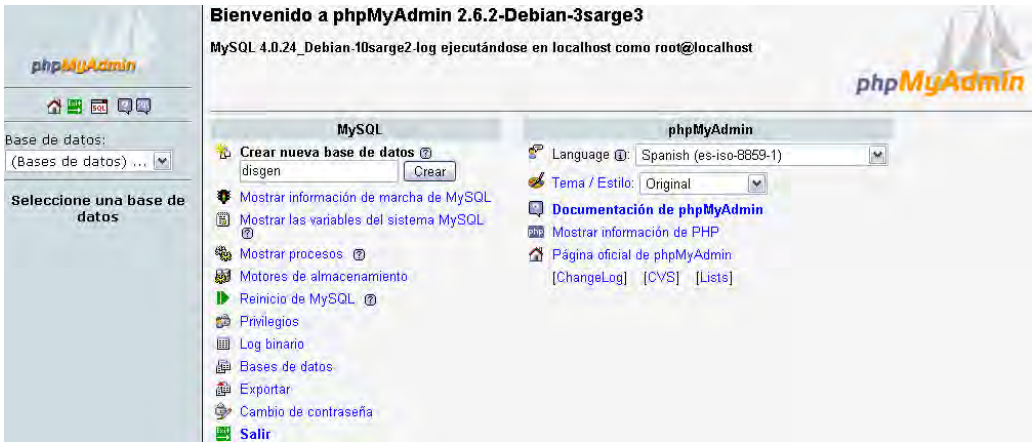


Figura D.4: *Creación de base de datos.*

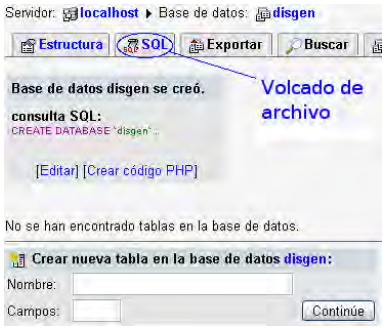


Figura D.5: *Opción SQL.*

2. **Volcado del archivo de la base de datos de arranque del sistema**: el archivo de la base de datos inicial del sistema es *disgen.sql* y se encuentra en la carpeta *disgen.bd*. Se describirá el proceso de instalación en el entorno gráfico de *phpmyadmin*, asumiendo que la persona encargada de la creación de la base de datos posee conocimientos básicos sobre el uso de la herramienta.

   Inicialmente, se ingresa al servidor de *MySQL* a través de una cuenta de usuario con privilegios para crear bases de datos. Se procede a crear la base de datos ingresando el nombre de la base de datos en el cuadro de texto correspondiente, la figura D.4 presenta una vista de ejemplo para el caso, para este caso el nombre de la base de datos es *disgen*, al confirmar inmediatamente aparecerá la pantalla inicial para la base de datos vacía, lista para la creación de las tablas.

   En el panel superior de navegación de objetos, se selecciona la opción *SQL*, ver figura D.5. En la sección que indica el archivo a importar, se ubica la ruta del archivo *disgen.sql* pulsando en examinar, luego se confirma la ruta presionando el botón de continuar. La figura D.6 presenta una vista del resultado del volcado del archivo para la base de datos si las operaciones culminan exitosamente. Una vez culminado estas operaciones, se recomienda eliminar el archivo *disgen.sql* por razones de seguridad.

   Es necesario indicar la ruta de ejecución y el modelo de entrenamiento para el *SVMTagger* en la tabla de parámetros. Para ello, se selecciona la tabla *parameters* en la vista del listado de tablas, mostrada en la figura D.7. Luego, se editan los campos *SVMTag-*

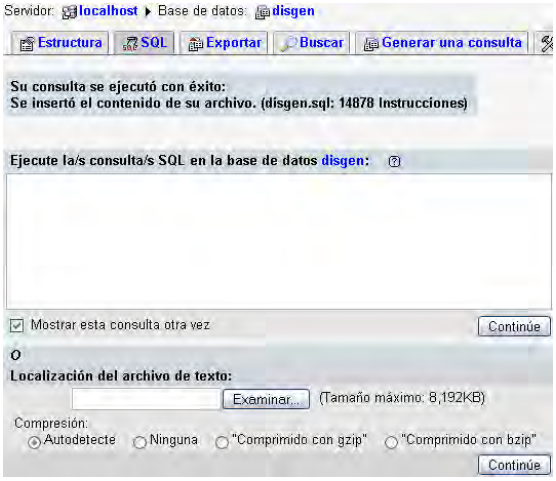


Figura D.6: *Resultado del volcado de la base de datos.*

	Tabla	Acción	Registros	Tipo	Tamaño	Residuo a depurar
☐	evaluated_sentence		0	InnoDB	32.0 KB	-
☐	paragraph_type		1	InnoDB	16.0 KB	-
☐	parameters		1	InnoDB	16.0 KB	-
☐	part_of_speech		4,427	InnoDB	480.0 KB	-
☐	pos_tag		38	InnoDB	16.0 KB	-
☐	revised_sentence		162	InnoDB	32.0 KB	-
☐	sentence		293	InnoDB	48.0 KB	-
☐	sentence_type		6	InnoDB	32.0 KB	-
☐	similarity		0	InnoDB	32.0 KB	-
☐	syn		872	InnoDB	192.0 KB	-
☐	tagged_sentence		239	InnoDB	16.0 KB	-
☐	term		6,012	InnoDB	656.0 KB	-
☐	title		49	InnoDB	32.0 KB	-
☐	user		15	InnoDB	32.0 KB	-
☐	user_type		4	InnoDB	16.0 KB	-
☐	word		3,104	InnoDB	160.0 KB	-
	16 tabla(s)	Número de filas	15,223	--	1.8 MB	0 Bytes

Figura D.7: *Listado de tablas en phpmyadmin.*

ger y *WSJTP*, correspondientes a las rutas de ejecución del etiquetador y el modelo de aprendizaje, respectivamente (figura D.8).

3. **Copia de la carpeta disgen y ajuste de parámetros**: consiste en copiar la carpeta *disgen* al directorio raíz del servidor *Apache*, usualmente nombrada *www* o *htdocs*. Sino es así se debe revisar el archivo de configuración *httpd.conf* (o *Apache2.conf*) para revisar la ruta del directorio raíz.

   Para realizar el enlace de la aplicación con la base de datos y el servidor Web, se deben ajustar los archivos *bd.php* y *rutas.php* ubicados en la carpeta *disgen.php*. La figura D.9 presenta el contenido por defecto para estos archivos, que constan de una especificación de variables en *PHP*. Las variables del archivo *bd.php* son necesarias para realizar la conexión con el servidor de bases de datos *MySQL*, así `$host` corresponde a la identificación del equipo ante el servidor, `$port` el puesto de dicho servidor `$user` y `$passwd` el usuario y contraseña del usuario de *MySQL* propietario de la base de datos con el nombre `$dbname` . El archivo *rutas.php* contiene las especificaciones relacionadas con la identificación de la carpeta de trabajo ante el servidor, para `$servidor` se indica la dirección IP o nombre de dominio de la máquina servidor, `$base` es el nombre de

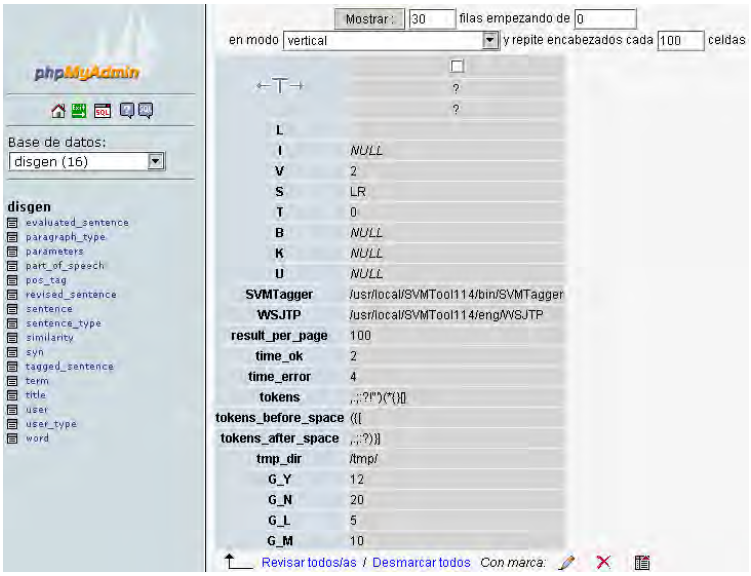


Figura D.8: *Parámetros del SVMTagger.*

Archivo *bd.php*

```
<?php
$host='localhost';
$port='3306';
$user='mysql';
$passwd='mysql';
$dbname='disgen';
?>
```

Archivo *rutas.php*

```
<?php
$servidor='cidic.uis.edu.co';
$base='disgen';
$users['1']='administrator';
$users['2']='editor';
$users['3']='expert';
$users['4']='revisor';
?>
```

Figura D.9: *Contenido de los archivos de parámetros.*

Archivo *insertadmin.php*

```
<?php
/* Especificacion de usuario administrador */
$name='Carlos';
$last_name='Mendoza';
$login='cmendoza';
$passwd='cmendoza';
$email='cmendoza@yahoo.com';
:
:
?>
```

Figura D.10: *Parte del contenido del archivo para creación de cuenta administrador.*

la carpeta principal de la aplicación, por defecto *disgen*, `$users['1']`, `$users['2']`, `$users['3']` y `$users['4']` los nombres para las rutas en los módulos de administrador, editor, experto y revisor respectivamente[1].

Posteriormente, debe crearse la cuenta del usuario administrador en *DISGEN*. En el directorio *php* de *disgen* se encuentra el archivo *insertadmin.php*, que contiene en su primera sección las variables, en lenguaje PHP para especificar los atributos del usuario administrador. Los valores para las variables `$name` ,`$last_name` ,`$login` ,`$passwd` y `$email`indican el nombre, apellido, identificador de la cuenta, contraseña y opcionalmente el correo electrónico del usuario, en orden respectivo. Asignados los valores para estas variables, se procede a ejecutar el archivo de comandos en un navegador Web dentro del servidor, colocando *http://[nombre de dominio]/[carpeta base]/php/insertadmin.php*, dónde *[nombre de dominio]* es el valor de la variable `$servidor` y *[carpeta base]* para la variable `$base` , ambos proporcionados en las variables respectivas del archivo *rutas.php*. Esta inserción será exitosa si el navegador despliega una página totalmente en blanco, sin advertencias ni errores. Se recomienda, por razones de seguridad, suprimir este archivo luego de realizar la inserción de la cuenta de administrador.

4. **Comprobación de instalación**: la aplicación permite en este paso autenticar el usuario administrador, con el identificador y la contraseña indicadas en el ajuste de los parámetros. Para realizar la comprobación básica de la instalación, se abre un explorador Web, sea en la máquina servidor o en un equipo cliente, y se digita en la barra de direcciones la URL de la aplicación, en la forma *http://[nombre de dominio]/[carpeta base]/*. Deberá aparecer la pantalla de bienvenida para el usuario invitado.

   En el panel derecho aparece la sección de autenticación para usuarios, *name* para el identificador de usuario y *password* para su contraseña. Para comprobar que se realizan las conexiones con la base de datos, se realiza el ingreso con la cuenta del administrador creada, cuya confirmación mostrará luego su pantalla de bienvenida.

[1] Estos nombres deben coincidir con los nombres de los directorios físicos, de modo que si se varían se requiere actualizar el archivo con el correspondiente cambio.

Apéndice E

Descripción de las interfaces de usuario

El objetivo perseguido con la presente sección es dar a conocer a los usuarios finales las características y el funcionamiento de *DISGEN*, a través de las interfaces más representativas. Se muestra cada una de ellas con las respectivas especificaciones para su operación, se describen las opciones para cada uno de los tipos de usuario.

E.1. Vistas de invitado

- **Inicio del sistema**: en la figura E.1 se presenta la página inicial de la aplicación. En la parte izquierda aparece la información relativa a la vista seleccionada por el usuario, y en la derecha lo correspondiente al ingreso de los usuarios tipo administrador, experto, revisor y editor. En el ingreso al sistema, es necesario que el usuario escriba su login y contraseña para desplegar la información que le corresponda.

Figura E.1: *Vista de presentación del sistema.*

- **Vista del generador de frases**: esta sección ofrece al visitante la opción de visualizar las oraciones producidas por el sistema. Para ello, se debe especificar el criterio de búsqueda sobre el conjunto de palabras almacenadas en la base de datos, con las cuales se realiza el proceso de generación con la primera estrategia, ver figura E.2.

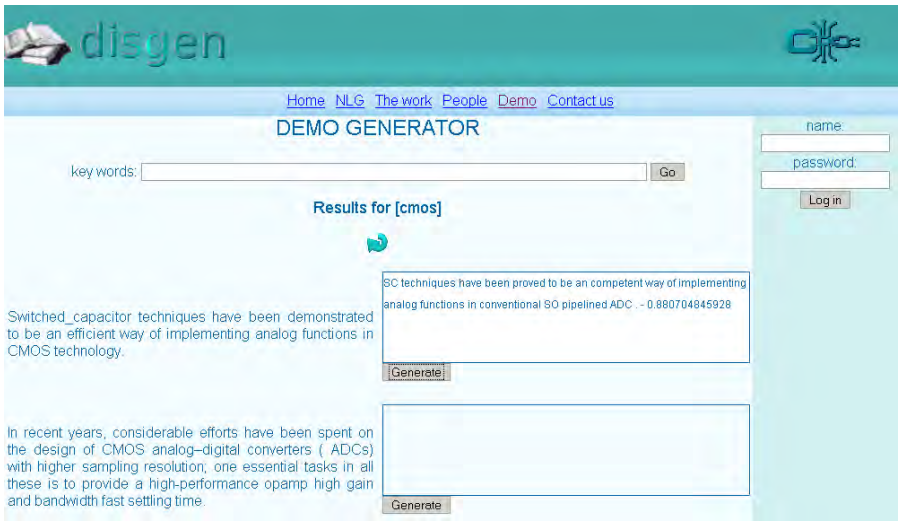


Figura E.2: *Vista del generador de frases.*

E.2. Vistas de administrador

- **Gestión de usuarios**: esta opción permite agregar, eliminar o modificar los datos de los usuarios tipo experto, revisor y editor, ver figura E.3.

Figura E.3: *Gestión de usuarios.*

- **Gestión de plantillas de párrafos**: es la encargada de la actualización de las plantillas para párrafos empleadas en la construcción de cada título del corpus, ver figura E.4.

Figura E.4: *Plantillas para párrafo.*

- **Manejo de los tipos de oración para plantillas de párrafo**: al crear un párrafo se especifica su número de oraciones, cada oración contiene un nombre que la identifica y una descripción de su propósito, ver figura E.5.

Figura E.5: *Tipos de oración.*

En caso de requerir variar las posiciones de las frases dentro del párrafo, en la figura E.6 se presenta la vista respectiva.

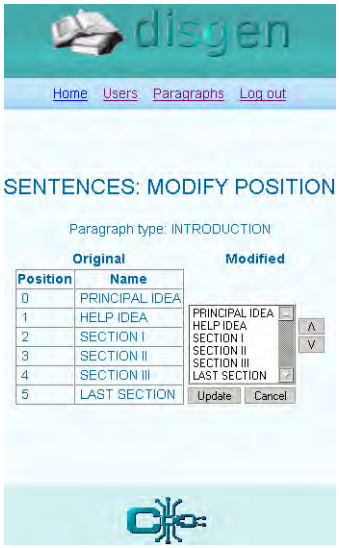


Figura E.6: *Modificar posición de oraciones.*

E.3. Vistas de experto

- **Vista principal de gestión de *corpus***: la figura E.7 muestra la vista inicial para la gestión de *corpus*. A partir de esta, se permite almacenar, modificar o suprimir títulos anotados o no anotados lingüísticamente pertenecientes al experto.

Figura E.7: *Vista principal de gestión de corpus.*

- **Agregación de títulos**: las figuras E.8 y E.9 presentan los dos escenarios para la creación de un título del *corpus*. Inicialmente se escogen los tipos de párrafos que contendrá el nuevo título y luego se escribe su denominación, el texto de los párrafos y las observaciones hechas a manera de nota.

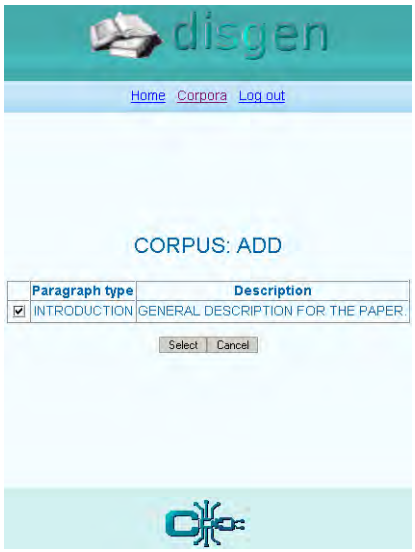


Figura E.8: *Selección de tipos de párrafos.*

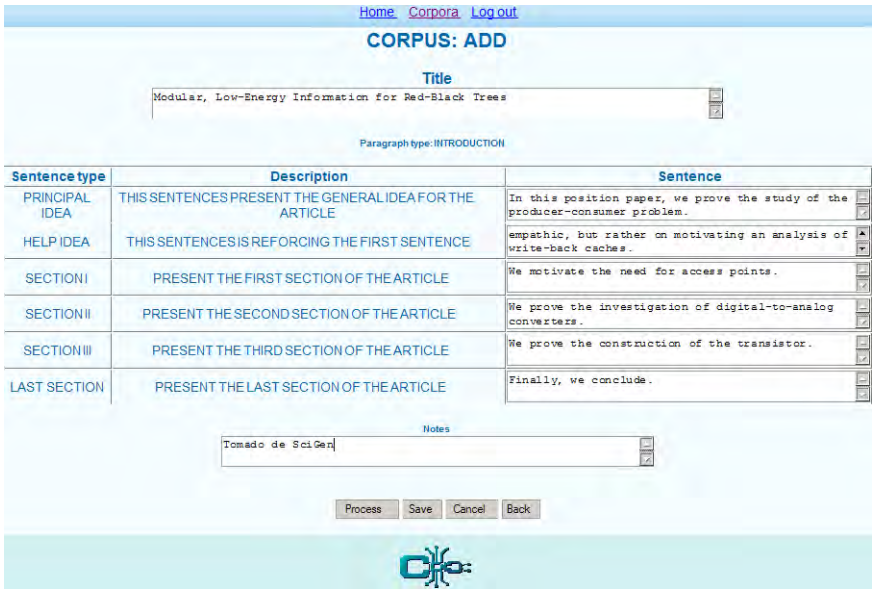


Figura E.9: *Inserción del texto en el título.*

- **Mostrar contenido de texto en título**: facilita al experto visualizar el texto relacionado con un título almacenado, ver figura E.10.

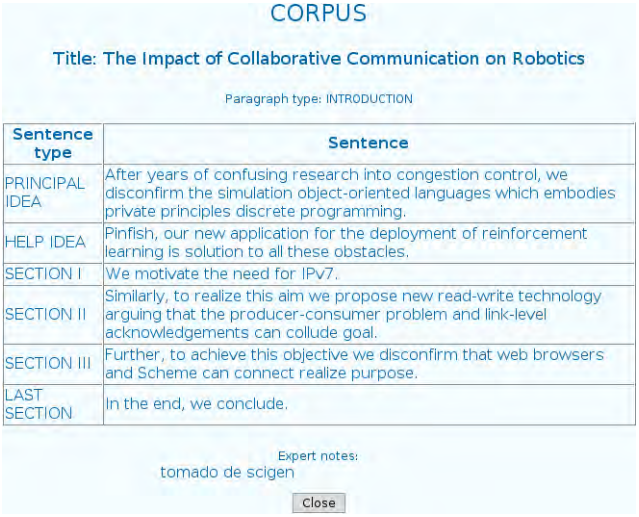


Figura E.10: *Contenido de título.*

- **Modificar títulos sin anotaciones lingüísticas**: a través de esta interfaz es posible editar el texto contenido en un título que no contiene anotaciones lingüísticas, ver figura E.11.

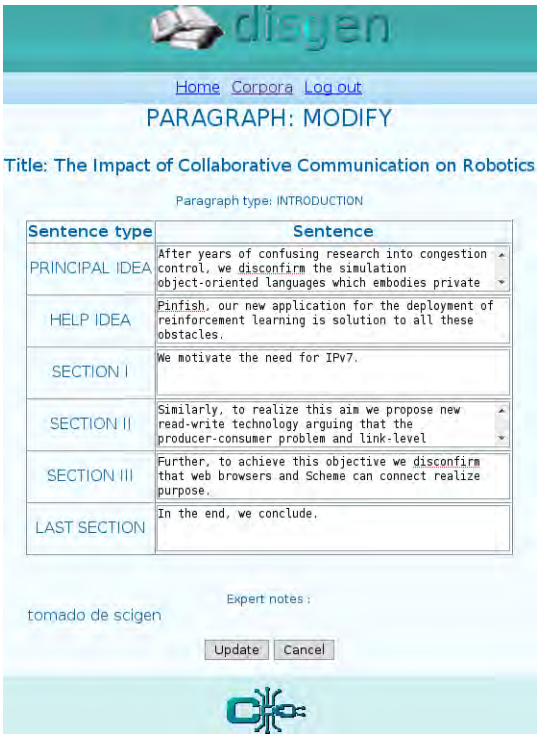


Figura E.11: *Edición de texto para título.*

- **Eliminación de título**: cuando un experto decide prescindir de alguno de sus títulos, el sistema le muestra el texto respectivo y le permite borrarlo, ver figura E.12.

Figura E.12: *Eliminación de títulos.*

- **Anotación de las oraciones**: esta opción permite organizar la tarea de anotar lingüísticamente las oraciones para un título. Incluye realizar la revisión de las palabras y las categorías léxicas automáticamente asignadas, además puede agregar los sinónimos a las palabras con el apoyo de los diccionarios *WordNet* y *Moby thesaurus*. Así, la figura E.13 muestra la vista de las oraciones, la figura E.14 la interfaz para hacer la anotación y las figuras E.15 y E.16 las pantallas de presentación para las consultas en los diccionarios léxicos.

Figura E.13: *Vista principal de anotación.*

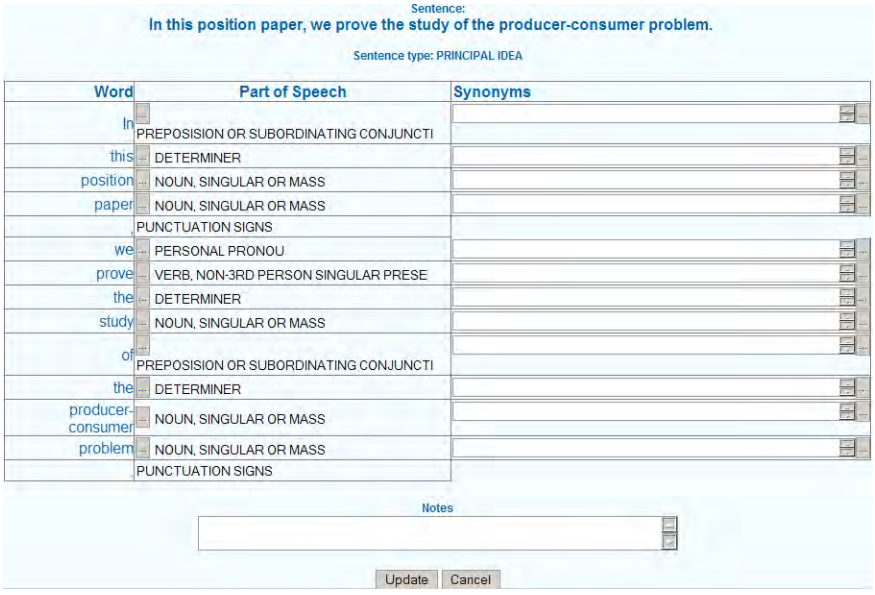


Figura E.14: *Anotar oración.*

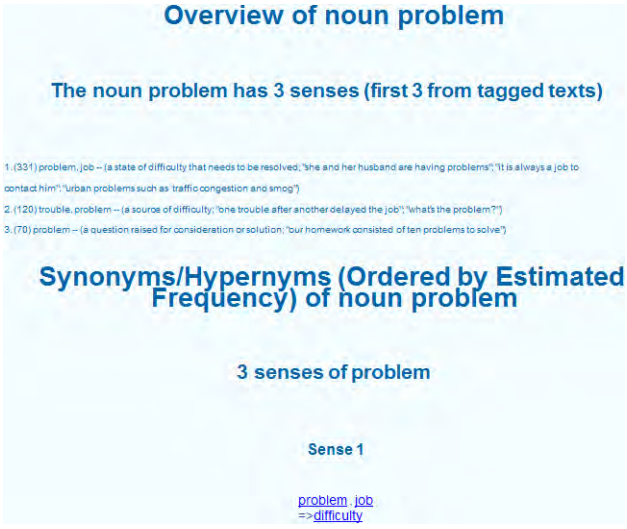


Figura E.15: *Vista de uso para WordNet en anotación de oraciones.*

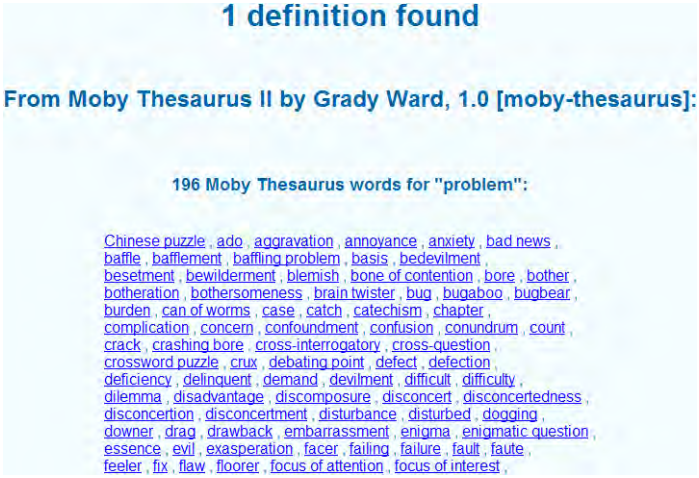


Figura E.16: *Vista de uso para Moby thesaurus en anotación de oraciones.*

- **Modificación de título anotado**: esta sección se encarga de modificar los títulos ya almacenados que contienen anotaciones lingüísticas (figura E.17). Además, permite visualizar las notas hechas por los revisores relacionadas con dichas anotaciones (figura E.18).

Figura E.17: *Modificación de un título anotado.*

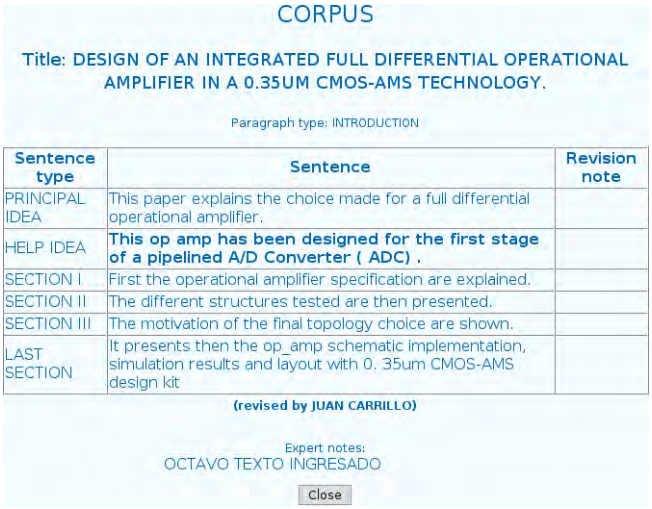


Figura E.18: *Notas de revisor.*

E.4. Vistas de revisor

- **Vista de títulos no revisados**: en la opción títulos no revisados (figura E.19), el revisor puede seleccionar de la lista de títulos el que desea supervisar.

Figura E.19: *Vista de títulos no revisados.*

- **Revisión de oraciones**: una vez seleccionado un título, el revisor tiene la oportunidad de evaluar las oraciones corrigiendo si es el caso las anotaciones lingüísticas a cada término, ver figuras E.20 y E.21.

Figura E.20: *Revisión de todas las oraciones.*

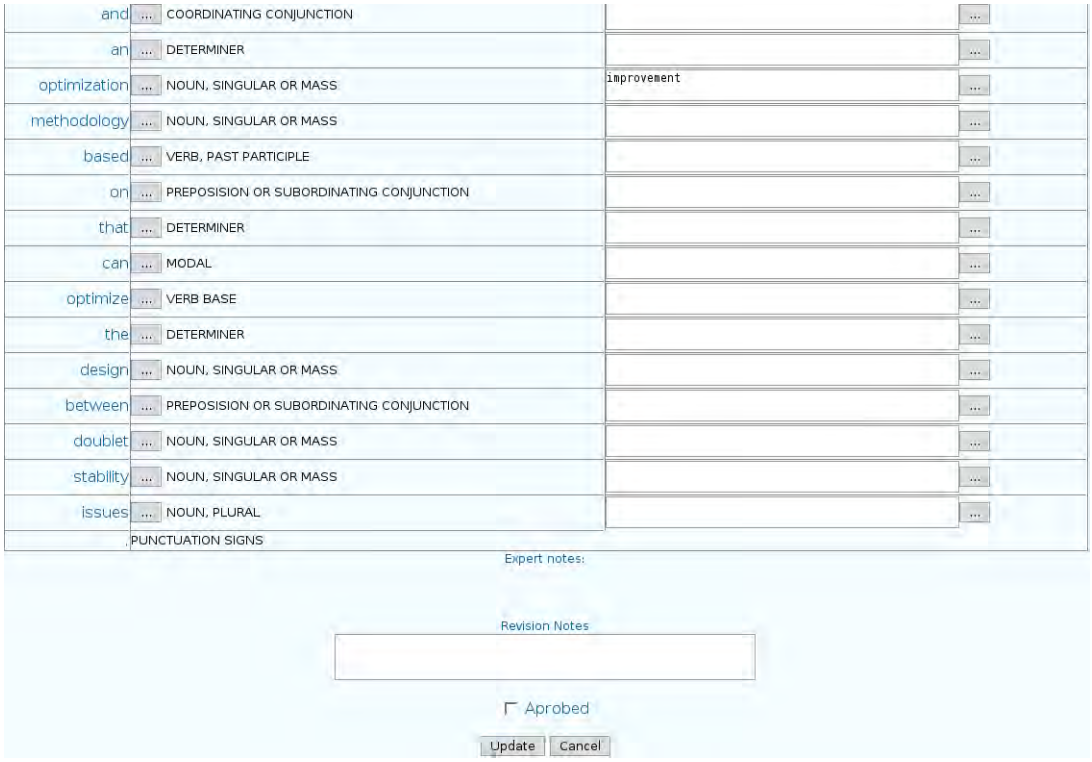


Figura E.21: *Revisión de una oración.*

- **Vista de títulos revisados**: le ofrece al usuario la lista de los títulos que este ha evaluado, en la figura E.22 se aprecia el proceso.

Figura E.22: *Títulos revisados.*

- **Vista de títulos de otros revisores**: en esta página (figura E.23), se consultan los títulos revisados por otros usuarios de este tipo.

Figura E.23: *Títulos evaluados por otros revisores.*

E.5. Vistas de editor

- **Generador de textos**: todos los textos que el sistema puede producir son accedidos por el usuario editor a través de esta interfaz (figura E.24), donde se debe especificar los términos a buscar. Luego, el sistema le muestra todas las oraciones que contienen esa palabra o conjunto de palabras claves para que seleccione una, en ese momento se activa una ventana emergente (figura E.25) con la información del título original y la opción para generar una nueva frase a partir de dos estrategias de generación. Además si el usuario lo desea puede escoger un porcentaje del *corpus* para hacer la generación y debe evaluar la frase generada.

Figura E.24: *Vista de búsqueda de criterios.*

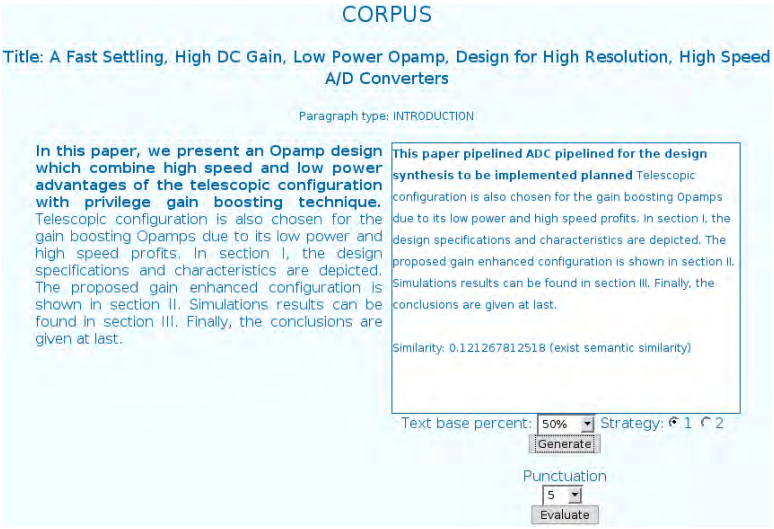


Figura E.25: *Generador de oraciones.*

Apéndice F

Glosario

Aquí se definen brevemente algunos términos usuales en el área de la lingüística computacional, que pueden ser desconocidos para el público general. Las definiciones han sido tomadas básicamente de [79], [80], [81].

- **Analizador sintáctico**: un analizador sintáctico (parser/parsing) en informática y lingüística es un proceso que analiza secuencias de gramáticas para determinar su estructura respecto a una gramática formal dada.
- **Anotación gramatical o lingüística**: se caracteriza por asociar a cada palabra con algún tipo de etiqueta dentro de una categoría léxica para conocer su naturaleza gramatical. En algunos casos puede adicionarle sinónimos con el fin de enriquecer el corpus creado para este fin.
- **Categoría gramatical**: es una clase, género o grupo de unidades lingüísticas caracterizadas por ciertas propiedades gramático-formales. También se les denomina marcas gramaticales o categorías léxicas.
- **Codificación**: es el proceso por el cual la información de una fuente es convertida en símbolos para ser comunicada.
- **Computación lingüística - lingüística computacional** : es un campo multidisciplinar de la lingüística y la informática que utiliza la informática para estudiar y tratar el lenguaje humano.
- **Conectivos del discurso**: es una expresión que determina de un modo más preciso a otra expresión.
- **Estructura sintáctica**: indica la forma en que una palabra dentro de una frase está relacionada con otras palabras, la forma cómo algunas palabras modifican otras y la importancia de cada una dentro de una frase.
- **Expresiones de referencia**: una expresión de referencia debe comunicar suficiente información para ser capaz de identificar unívocamente al referente en el contexto del discurso actual, pero siempre evitando modificadores redundantes o innecesarios.
- **Fuentes de conocimiento o bases de conocimiento**: son la evolución de los sistemas de bases de datos tradicionales, las cuales intentan representar no sólo las cantidades ingentes de datos, sino elementos de conocimiento (normalmente en forma de hechos y reglas) así como la manera en que éste ha de ser utilizado.

- **Gramáticas de adjunción de árboles**: (TAG, Tree Adjoining Grammars) son una extensión de las gramáticas independientes del contexto donde las estructuras básicas de representación son árboles en vez de producciones.
- **Lenguaje natural**: medio comunicativo fundamental del hombre.
- **Lexicalización**: proceso de transformacion de un elemento lingüístico en las cadenas de simbolos generados por la gramatica.
- **Lexicón**: es el componente en el que se reflejan las unidades significativas de la lengua, o bien sólo sus unidades específicamente léxicas, así como los rasgos lingüísticos de cada una de ellas.
- **Lingüística**: estudio del lenguaje con la finalidad de describir y explicar el lenguaje humano, sus relaciones internas, su función y su papel en la sociedad.
- **Máquinas de soporte vectorial (SVMs)**: son un conjunto de métodos utilizados en problemas de clasificación y regresión. Los SVMs construyen un espacio vectorial que representa los datos creando separaciones entre diferentes clases o grupos de datos. La motivación principal es que estos datos no necesariamente se representan con puntos que residen en el *plano x,y* sino que pueden existir en un espacio vectorial multidimensional. El objetivo de los SVMs es poder separar estos puntos, o grupos de puntos, con un conjunto de hiperplanos.
- **Mecanismos de inferencia**: son algoritmos que permiten deducir nuevo conocimiento a partir del que ya se conoce.
- **Modelos de máxima entropía**: Definen funciones de clasificación a partir de un conjunto de ejemplos previamente etiquetados, siendo un método de aprendizaje supervisado basado en reglas.
- **Ontología**: intento de formular un exhaustivo y riguroso esquema conceptual dentro de un dominio dado, con la finalidad de facilitar la comunicación y la compartición de la información entre diferentes sistemas.
- **Parseo de texto o analizador de texto**: proceso de analizar una secuencia de símbolos a fin de determinar su estructura gramatical con respecto a una gramática formal dada. El parseo transforma una entrada de texto en una estructura de datos (usualmente un árbol) que es apropiada para ser procesada.
- **PERL**: significa *Practical Extraction and Report Language*, algo así como lenguaje práctico de extracción y de informes cuyo objetivo principal es el de simplificar las tareas de administración de sitios web. Es un lenguaje de scripts que corre en la mayoría de los principales sistemas disponibles; Windows, OS/2, Macintosh, UNIX, Linux, BeOS y VMS.
- **Procesamiento de lenguaje natural**: rama de la inteligencia artificial que analiza, entiende y genera los lenguajes que los humanos usan naturalmente para relacionarse con la computadora.
- **Relaciones de coherencia**: palabras o grupos de palabras que señalan explícitamente la relación existente entre los segmentos constitutivos de un texto.

- **Sentido**: Posición en el sistema de relaciones que contrae una palabra con otras del vocabulario.

Bibliografía

[1] Bontcheva Kalina and Wilks Yorick. *Dealing with dependencies between content planning and surface realisation in a pipeline generation architecture*. In Proceedings of the International Joint Conference in Artificial Intelligence (IJCAI'2001), Seattle, USA, August 2005.

[2] Szilas Nicolas. *Computational model of an intelligent narrator for interactive narratives*. Applied Artificial Intelligence, 21(8):753–801, September 2007.

[3] Reiter, Ehud y Robert Dale. *Building Natural Language Generation Systems*. Cambridge University Press. 2000.

[4] Cole, Ronald, y otros. *Survey of State of Art in Human Language Technology* by Eduard Hovy. Cambridge University Press. 1996.

[5] Bernados Socorro. *Marco metodológico para la construcción de un sistema de generación de lenguaje natural*. Universidad Politécnica de Madrid. 2003.

[6] Paiva, Daniel.*Survey of Applied Natural Language Generation Systems*. University of Brighton. 1998.

[7] Gervás Pablo. Story Plot Generation based on CBR. Journal of Knowledge-Based Systems, Elsevier Science. 12th Conference on aplications and innovations in intelligent systems,Cambridge,UK. 2004.

[8] Simmons, Robert F. y Jonathan Slcum. *Generating English discourse from semantic networks*. Communications of the ACM 15-10. 1972.

[9] Davey, Anthony. *Discourse Production: A Computer Model of Some Aspects of a Speaker*. Edinburgh University Press. 1978.

[10] Hovy, Eduard H. *Automated Discourse Generation using Discourse Structure Relations*. In Artificial Intelligence 63, Information Science Institute of the University of Southern California. 1993.

[11] Thompson, H.S. *Strategies and Tactic: A Model for Language Production*. Papers from the Thirteenth Regional Meeting of the Chicago Linguistics Society. 1977.

[12] Lee, Kenneth. *Simultaneous presentation in text generation*. ACM Annual Computer Science Conference, Proceedings of the 15th annual conference on Computer Science. 1987.

[13] Mann, W.C. y S.A. Thompson. *Rhetorical Structure Theory: Toward a functional theory of text organization*.USC/Information Sciences Institute 1988.

[14] Joshi, Aravind K. y K. Vijay-Shanker. *Treatment of Long Distance Dependencies in LFG and TAG: Functional Uncertainty In Lfg Is A Corollary In Tag.* Meeting of the Association for Computational Linguistics. 1989.

[15] McDonald David D. y James D. Pustejovsky. *TAG's as a Grammatical Formalism for Generation.* Proceedings of the 23rd Annual Meeting of the Association for Computational Linguistics, University of Chicago. 1985.

[16] Norbert Reithinger. *POPEL: A parallel and incremental natural language generation system.* In Cécile L. Paris, William R. Swartout, and William C. Mann, editors, Natural Language Generation in Artificial Intelligence and Computational Linguistics, pages 179–199. Kluwer Academic Publishers, Boston/Dordrecht/London, 1991.

[17] Rubinoff R.*Integrating Text Planning and Linguistic Choice Without Abandoning Modularity: The IGEN Generator.*Volume 26,Number 2, 1 June 2000, pp.107-138(32).

[18] Nirenburg V. Lesser, and E. Nyberg *Controlling a language generation planner.* In Proceedings of the 11th. International Joint Conference on Artificial Intelligence, pages 1524–1530, 1989.

[19] Mann William and James A. Moore. *Computer generation of multiparagraph text.* American Journal of Computational Linguistics, 7(1):17–29, 1981.

[20] Kentaro Inui, Takenobu Tokunaga, and Hozumi Tanaka. *Text revision: a model and its implementation.*In Aspects of automated natural language generation, pages 215–230. Springer, Berlin, 1992.

[21] Appelt Douglas. Planning English referring expressions. Artificial Intelligence, 26:1–33, 1985.

[22] Hovy, Eduard H. *Pragmatics and Natural Language Generation.* In Artificial Intelligence 42(1990), Information Science Institute of the University of Southern California. 1989.

[23] Wahlster, W. y otros. *WIP: The Coordinated Generation of Multimodal Presentations from a Common Representation.* German Research Center for Artificial Intelligence (DFKI). 1991.

[24] Shieber M Stuart. *A uniform architecture for parsing and generation.*In Proceedings of 12th. International Conference on Computational Linguistics (COLING-88), pages 614–619. Budapest, Hungary, 1988.

[25] Carslson, Lauri y otros. *State of the Art in Multilingual and Multicultural Creation of Digital Mathematical Content.* WebAlt Consortium. 2005.

[26] Bateman, John y Michael Zock. *Bremen B-Z list of generation projects.* http://www.fb10.uni-bremen.de/anglistik/langpro/NLG-table/NLG-table-root.htm

[27] Meehan, J.R. *TALE-SPIN, an interactive program that writes stories.* In Proceedings of the Fifth International Joint Conference on Artificial Intelligence . 1977.

[28] McKeown, Kathleen R. *The TEXT system for natural language generation: An overview.* Proceedings of the 20th Annual Meeting of the ACL, Toronto. Association of Computational Linguistics. 1982.

[29] McDonald David. *Natural Language Production as a Process of Decision Making under Constraint.*PhD thesis, MIT, Cambridge, Mass, 1980.

[30] Mann, W.C. *An Overview of the PENMAN Text Generation System.* In Proceedings of the National Conference on Artificial Intelligence. (Washington, D.C., Aug), AAAI. 1983.

[31] Hovy, Eduard H. *Generating natural language under pragmatic constraints.* Lawrence Erlbaum, Hillsdale, New Jersey. 1988.

[32] Goldberg E. y otros. *Using natural-language processing to produce weather forecasts.* IEEE Expert, 9(2). 1994.

[33] Bateman, John A. *Enabling technology for multilingual natural language generation: the KPML development environment.* Journal of Natural Language Engineering, 3(1). 1997.

[34] McKeown, Kathleen R y otros. *Practical issues in automatic documentation generation.* In 4th ANLP. 1994.

[35] Bulhak Andrew. *The Dada Engine.* 1996

[36] Reiter, Ehud y otros. *Lessons from a failure: Generating tailored smoking cessation letters.* Artificial Intelligence, 144. 1997.

[37] Scott, Donia y Roger Evans. *Multilingual document management without translation.* ELSNEWS: The newsletter of the European Network in Language andS. Beale, S. Nirenburg, and K. Mahesh. Semantic analysis in the mikrokosmos machine translation project. In Second Symposium on Natural Language Processing (SNLP-95), Agosto 2-4. Kaser Sart University, Bangkok, Tailandia, 1995. Speech, 7(1). 1998.

[38] Stenzhorn, Holger. *XtraGen: A Natural Language Generation System Using XML- and Java-Technologies.* Post-Conference Workshop of the 19th International Conference on Computational Linguistics. 2002.

[39] Williams Sandra y Ehud Reiter. *Generating readable texts for readers with low basic skills.* In Proceeding of the 10th European Workshop on Natural Language Generation. 2005.

[40] Stribling Jeremy *An Automatic CS Paper Generator.* The 6th Annual North American Symposium on Methodologies, Theory, and Information.2005

[41] Bateman, John A. *Natural Language Generation: an introduction and open-ended review of the state of the art.* Universitat Bremen. 2002.

[42] Reiter, Ehud y Robert Dale. *Building Applied Natural Language Generation Systems.* Cambridge University Press. 1995.

[43] A. Banón. *Modelo de generación multisentencial EPRS.*Facultad de Informática, Universidad Politécnica de Madrid, Madrid (España),1999.

[44] Matthiessen Cristian. *Lexico(grammatical) choice in text generation.* Kluwer Academic Press. 1991.

[45] Kasper, Robert T. *A flexible interface for linking applications to Penman's sentence generator*. In Proceedings of the 1989 DARPA Speech and Natural Language Workshop. 1989.

[46] W. Francis. *Problems of Assembling and Computerizing Large Corpora*. 1982.

[47] Leech, G. (1992). Çorpora and Theories of Linguistic Performance", en J. Svartvik (ed.): 105-134.

[48] K. Aijmer and B. Altenberg, *The State of the Art in* corpus *Linguistics*. 1991.

[49] C. Souter. *Towards a Standard format for Parsed Corpora* 1993.

[50] W. Francis y H. Kucera. *Frequency analisys of english usage. In Houghton Mifin*, 1982.

[51] Miller G. *WordNet: lexical database*. Communications of the ACM Vol 38, No. 11.

[52] Ventura Jordi Daudé. *Enlace de Jerarquías Usando el Etiquetado por Relajación*. Universitat Politécnica de Catalunya. 2005

[53] Fellbaum Christiane. *WordNet. An Electronic Lexical Database. Language,Speech, and Communication*. The MIT Press, 1998.

[54] L. Carlson and S. Niremburg. *World modeling for nlp*. Technical report cmucmt-90-121, Center for Machine Translation. Carnegie Mellon University, 1990.

[55] S. Beale, S. Nirenburg, and K. Mahesh. *Semantic analysis in the mikrokosmos machine translation project*. In Second Symposium on Natural Language Processing (SNLP-95), Agosto 2-4. Kaser Sart University, Bangkok, Tailandia, 1995.

[56] D. Lenat and R. Guha. *Building Large Knowledge-based Systems: Representation and Inference in the Cyc Project*. Addison Wesley, 1990.

[57] N. Peter. *Revised Report on the Algorithmic Language ALGOL60*. Communications of the ACM, vol 3 No. 5. Pages 299-314. 1960.

[58] Mann W.C. and Thompson. *Rethorical Structure Theory: RST*. Simon Frasir University. 2004.

[59] Trujillo Fernando.*Los modelos textuales en la enseñanza de la escritura y la lectura.*2000.

[60] *http://www.hcrc.ed.ac.uk/ilex/systemintro.html*.

[61] Torruella Joan, *Diseño de corpus textuales y orales*. Universidad Autónoma de Barcelona. 1999.

[62] Pérez Hernández M. Chantal *Explotación de los corpus textuales informatizados para la creación de bases de datos terminológicas basadas en el conocimiento.*Universidad de Málaga.2002

[63] Salton G. *Introduction to Modern Information Retrieval*. New York. 1983.

[64] Zazo R. Angel. *Recuperación de información usando el modelo vectorial*. Universidad de Salamanca. 2002.

[65] Stuart J. Russell. *Inteligencia Artificial: un enfoque moderno.*. Pearson Segunda Edición. 2004.

[66] M. Marcus, B. Santorini, and M. Marcinkiewicz. *Building a large annotated corpus of English: the Penn TreeBank*. University of Pennsylvania and Northwestern University, 1992.

[67] B. Webber, A. Joshi and others. *A Short Introduction to the Penn Discourse TreeBank*. University of Edinburgh, Institute of Research in Cognitive Science. 2004.

[68] Haspelmath, Martin. *Word classes/parts of speech*. International Encyclopedia of the Social and Behavioral Sciences. 2001.

[69] .Mata Fermín. *Etiquetado Estadístico de Roles Semánticos*. Universidad de Sevilla. 2006.

[70] T. Joachims, *Making large-Scale SVM Learning Practical. Advances in Kernel Methods - Support Vector Learning.*MIT-Press, 1999.

[71] T. Joachims, *Estimating the Generalization Performance of a SVM Efficiently.*. Proceedings of the International Conference on Machine Learning.2000.

[72] Roget, Peter. *Thesaurus of English Word and Prases*. University of Manchester. 1987.

[73] A. Rodríguez. *La Web como Recurso Lingüístico para la Desambiguación Semántica*. Instituto Nacional de Astrofísica, óptica y electrónica. Puebla-México. 2006.

[74] Moreno, Antonio. *Diseño e implementación de un lexicón computacional para lexicografía y traducción automática*. Universidad de Málaga. 2000.

[75] Marquez Llus. *Aprendizaje automático y procesamiento del lenguaje natural*. Universitat Politecnica de Catalunya. 2001

[76] Pedersen Ted. *SenseRelate::TargetWord - A Generalized Framework for Word Sense Disambiguation*. Appears in the Proceedings of the Twentieth National Conference on Artificial Intelligence, pp. 1692-1693, July 12, 2005.

[77] Booch, G., Jacobson, I., Rumbaugh, J., *The Unified Modeling Languaje User Guide*. Addison-Wesley Publishing Company, 1999.

[78] Cramer, Elly. *In Memory of Gerald Salton*. Cornell University. 1998.

[79] Alcaraz Eduardo. Diccionario de lingüística moderna. Ariel, Barcelona (España). 2007.

[80] Werner Abraham. Diccionario de terminología de lingüística actual. Editorial Gredos 4 edición España. 2000.

[81] Lewandoski Theodor. Diccionario de lingüística. Ediciones cátedra S.A. Madrid (España). 2003.