Revista Signos
ISSN: 0035-0451
revista.signos@ucv.cl
Pontificia Universidad Católica de Valparaíso
Chile

Ferreira, Anita; Kotz, Gabriela
ELE-Tutor Inteligente: Un analizador computacional para el tratamiento de errores gramaticales en Español como Lengua Extranjera
Revista Signos, vol. 43, núm. 73, 2010, pp. 211-236
Pontificia Universidad Católica de Valparaíso
Valparaíso, Chile

Revista Signos
2010, 43(73)
211-236

ELE-Tutor Inteligente: Un analizador computacional para el tratamiento de errores gramaticales en Español como Lengua Extranjera*

Anita Ferreira
Gabriela Kotz
Universidad de Concepción
Chile

Resumen: Un Sistema Tutorial Inteligente (STI) está basado en los desarrollos de la Inteligencia Artificial, principalmente, en lo que compete a la utilización de técnicas de comprensión de lenguaje natural o de generación de lenguaje. Un STI es un programa para la enseñanza-aprendizaje basado en el computador cuya finalidad última es la facilitación de los procesos de aprendizaje personalizados. La etiqueta inteligente se refiere a la capacidad del sistema para analizar gramaticalmente la entrada y luego generar un *feedback* adecuado para el error del estudiante. Para ello, se deben implementar técnicas de procesamiento de lenguaje natural que se basan en teorías gramaticales para procesar la entrada del estudiante con el objeto de generar una estrategia de *feedback*. En materia de enseñanza de lenguas extranjeras, se ha corroborado empíricamente el interés en el aprendizaje que despierta por parte de los alumnos la incorporación de plataformas *e-learning* habilitadas con tecnologías de información y comunicación (TICS). Sin embargo, al mismo tiempo, se ha observado las limitaciones de dichas plataformas en el tratamiento gramatical específicamente en el reconocimiento de los diferentes tipos de errores de lengua (gramaticales, léxicos, etc.) que cometen los alumnos y la precariedad y poca efectividad en el tipo de retroalimentación en el tratamiento de dichos errores. Los errores y el *feedback* correctivo son parte natural del aprendizaje de la lengua. En este artículo se plantea como objetivo principal atender a la problemática de mejorar los procesos de aprendizaje del ELE, a través del Diseño e implementación de un Analizador Sintáctico Computacional de un STI que permita identificar y clasificar los errores gramaticales en Español como LE.

Palabras Clave: Español como Lengua Extranjera, Sistemas Tutoriales Inteligentes, *parser,* procesamiento de lenguaje natural.

* La investigación que se presenta en este artículo se circunscribe en el contexto del Proyecto FONDECYT 1080165. *Modelo Blended Learning basado en el enfoque por tareas y aprendizaje cooperativo para la enseñanza del Español como Lengua Extranjera*. CONICYT-CHILE.

Recibido:
5-I-2010
Aceptado:
25-III-2010

Correspondencia: Anita Ferreira (aferreir@udec.cl). Departamento de Español, Facultad de Humanidades y Arte, Universidad de Concepción. Casilla 160-C, Correo 3, Concepción, Chile.

ELE-Intelligent Tutor: A computational parser for the processing of grammatical errors in Spanish as a Foreign Language

Abstract: Intelligent Tutorial Systems (ITS) are based on the advances in Artificial Intelligence, mainly regarding the use of comprehension techniques for natural language or natural language generation. An ITS is a computer-based programme for teaching and learning and aims to facilitate personalized learning processes. The label of 'intelligent' is associated with the system's capacity to analyse the grammar of the language entry and then generate the appropriate feedback according to the student's error. To do this, it is necessary to use natural language processing techniques based on grammar-related theories so that the student's entry can be processed to generate a specific feedback strategy. The interest in learning Spanish as a Foreign Language has increased over the years, and this has resulted in the elaboration of learning environments highly improved by professionals and researchers that effectively support the processes to foster student learning autonomy. As to foreign language learning, the interest that students have in the incorporation of e-learning platforms empowered with Information and Communications Technologies (ICTs) has been empirically proven. However, the limitations of such platforms have also been observed regarding the processing of linguistic forms, specifically in terms of recognizing the different language errors that students make (grammar, lexicon, etc.) and the precariousness and lack of effectiveness in the type of feedback used to process these errors. Corrective feedback and errors are an essential and normal part of the language learning process. The main objective of this article is to meet the need of improving the Spanish as a foreign language learning process through the design and implementation of the parser of an intelligent tutorial system that could identify and classify grammatical errors in Spanish as a foreign language.

Key Words: Spanish as a Foreign Language, Intelligent Tutorial Systems, parser, Natural Language Processing.

INTRODUCCIÓN

Los Sistemas Tutoriales Inteligentes (STI) para Lenguas Extranjeras (LE) (del inglés, *Intelligent Tutorial Systems* (ITS) *for Foreign Language* (FL)) reciben el nombre de inteligentes porque son capaces de comportarse como un experto, ya que pueden analizar la situación en la que se encuentra el estudiante y proporcionar una solución acorde a la problemática. Para ello, se deben implementar técnicas de procesamiento de lenguaje natural (NLP) que se basan en teorías gramaticales para procesar la entrada del usuario con el objeto de generar una respuesta (Graesser, Person, Harter & TRG, 2001; Graesser, Jeon, Yang & Cai, 2007). Con el objeto de diseñar un sistema tutorial inteligente, es decir, un sistema que incorpore técnicas de NLP que sea capaz de reconocer de manera efectiva los errores gramaticales que efectúen los alumnos y dar una respuesta útil y específica, es necesario construir un analizador sintáctico *(parser)* para el tratamiento de dichos errores que opere en un entorno virtual y que dé respuestas automáti-

cas. La particularidad de este analizador gramatical de entradas (*parser*) es que debe procesar entradas erróneas y para ello es necesario poder predecir los errores que puede cometer un usuario en un momento determinado de su aprendizaje y en un tópico gramatical específico.

Este estudio se plantea como objetivo principal el diseño y la implementación de un analizador automático de errores gramaticales para el español en el contexto de un tutorial inteligente para la enseñanza del Español como Lengua Extranjera: *Parser* ELE-UDEC. El artículo está organizado en las siguientes secciones: En la sección 1, nos referimos a los principales fundamentos teóricos y empíricos en que se sustenta nuestra investigación. En la sección 2, presentamos el diseño e implementación del analizador automático para la enseñanza del Español como Lengua Extranjera. En la sección 3, mostramos algunos ejemplos del análisis sintáctico computacional llevado a cabo por el *Parser* ELE-UDEC. Finalmente, en la sección 4, se presentan comentarios finales.

1. Fundamentación teórica y empírica

1.1. ¿Qué es un analizador automático?

Un analizador automático (*parser)* de lenguaje natural toma un texto en lengua escrita como entrada y produce una representación formal de la estructura sintáctica (y, a veces semántica) del mismo, generalmente en forma de árboles. Los *parsers* diseñados para la enseñanza de lenguas generalmente contienen un componente que anticipa o detecta los errores en el caso de que las reglas gramaticales sean violadas. Por ejemplo, un procedimiento muy común para la detección de errores es la incorporación de una gramática de errores con *buggy rules* (reglas agramaticales), lo que permite que el *parser* procese una oración que contiene uno o más errores e identificarlos.

1.2. Efectividad de los parsers

La efectividad de los *parsers* sintácticos en la enseñanza de lenguas fue analizada empíricamente. Jouzulynas (1994, cit. en Heift & Schulze, 2007) evaluó la utilidad de los *parsers* sintácticos en el diagnóstico de errores. Para ello, analizó errores en un corpus de aproximadamente 400 páginas de ensayos en alemán escritos por estudiantes norteamericanos en cursos de segundo año de lengua. Su estudio demuestra que el área de aprendizaje más problemático es la sintaxis, seguido por la morfología. El estudio, que fue realizado con la ayuda de un *parser* del alemán, indicó que la mayoría de los errores de los alumnos (80%) no son de origen semántico y, por lo tanto, pueden ser reconocidos por un *parser* sintáctico.

Holland, Kaplan y Sams (1995) responden a la pregunta *¿Para qué sirven los parsers?* basados en los excelentes resultados obtenidos con BRIDGE, un sistema de CALL (del inglés, *Computer Assisted Language Learning)* basado en *parser* utilizado con personal militar norteamericano que aprendía alemán. Estas conclusiones son confirmadas con pruebas realizadas en otras aplicaciones de CALL basadas en *parser.* Por ejemplo, Nagata (1996) comparó sistemas de CALL con los sistemas de ICALL *(Intelligent-CALL*) o CALL basado en *parser,* y concluyó que ICALL resulta más eficiente, ya que realiza un uso más efectivo del computador, en el sentido de proveer una respuesta inmediata y apropiada para el alumno. Para ello utilizó el sistema Nihongo-CALI para el aprendizaje del japonés.

1.3. Procesamiento del lenguaje natural (del inglés NLP)

El procesamiento de lenguaje natural es una rama de la Inteligencia Artificial (AI), pero la AI no trata solo con el procesamiento de lenguaje natural. En general, la AI se ocupa de emular aspectos de la inteligencia humana a través de máquinas. La AI se ocupa de representaciones del conocimiento y la investigación y aplicación de tales algoritmos y técnicas en juegos, máquinas de aprendizaje, robótica. El NLP no solo es una importante rama de la AI, sino también de la Lingüística Computacional. Tradicionalmente se enfoca al desarrollo de HLT *(Human Language Technology)*, pero en la actualidad se ha dirigido más hacia al análisis detallado de problemas prácticos.

El procesamiento del lenguaje natural (del inglés, *Natural Language Processing,* NLP*)* se relaciona muchas veces erróneamente solo con la comprensión del lenguaje natural (del inglés, *Natural Language Understanding,* NLU), pero mientras que la comprensión del lenguaje natural incluye solo la interpretación del texto, el procesamiento de lenguaje natural atañe tanto a la comprensión como a la generación o producción de lenguaje (del inglés, *Natural Language Generation* (NLG) (Siddiqui & Tiwari, 2008). Lo que distingue el NLP de otros sistemas de procesamiento es el conocimiento del lenguaje. En efecto, el sistema debe conocer todos los niveles de la lengua (morfología, léxico, sintaxis), además de las convenciones de discurso y de uso (pragmática) (Jurafsky & Martin, 2000).

La tecnología NLP permite programar el computador con suficiente información lingüística en forma de reglas y patrones que permite realizar numerosas actividades relacionadas con el aprendizaje de lenguas. Por ejemplo, puede analizar estructuras de oraciones entregadas por los usuarios, detectar errores lingüísticos y, en algunos casos, aparentar comprensión entregando respuestas *ad hoc* o respondiendo a una conversación. Los sistemas con tecnología NLP les dan a los alumnos la factibilidad de crear oraciones nuevas y originales en la lengua que están

aprendiendo, ingresarlas al computador y recibir una respuesta como retroalimentación. Esto es lo que diferencia ICALL de CALL.

1.4. Técnicas básicas para el tratamiento de lenguaje natural

La mayoría de técnicas de procesamiento de lenguaje natural se desarrolla a través de etapas que pueden operar de manera secuencial o paralela.

1.4.1. Etapas de pre-procesamiento

La primera etapa de cualquier sistema de procesamiento de la lengua tiene lugar en el nivel textual. En este nivel, el texto puede ser considerado como una simple secuencia de caracteres. Las tareas básicas que deben abordarse a este nivel son: a) la segmentación del texto, b) el filtrado de información no relevante y, c) la localización de unidades tratables. A pesar de su aparente simplicidad, estas tareas pueden presentar algunas complicaciones.

a) Segmentación del texto. El texto debe ser segmentado en fragmentos tratables de manera automática. La dificultad de la tarea depende tanto de las características de los fragmentos a obtener (párrafos, oraciones, etc.), como de la fuente de la cual se obtienen (texto marcado, texto plano, resultado de una transcripción a partir de voz, etc.). Si se desea segmentar un texto en párrafos y se dispone de signos de puntuación y se puede distinguir mayúsculas y minúsculas, entonces la tarea es relativamente sencilla (aun cuando un signo de puntuación puede cumplir funciones diferentes de la de separación; por ejemplo, un punto puede formar parte de un nombre propio, de una sigla, de una fórmula o de un acrónimo). Si deseamos segmentar un texto en oraciones, la tarea es obviamente más difícil y será necesario algún tipo de conocimiento lingüístico, lo mismo que si la fuente a tratar es producto de una transcripción y carece de signos de puntuación y de distinción entre mayúsculas y minúsculas.

Esta tarea es realizada por un analizador léxico que crea estos segmentos, también llamados *tokens* y son estos *tokens* los que son procesados por el analizador morfológico, para posteriormente realizar el análisis sintáctico para construir luego la estructura de datos, por ejemplo, un árbol de análisis o árboles abstractos de sintaxis.

Antiguamente, había que tomar la cadena y separarla en cada elemento o carácter para tratarlo por separado y determinar si era el comienzo o el fin de un *token*, o un *token* en sí mismo. Era una tarea complicada y difícil de optimizar. En la actualidad existen en casi todos los lenguajes de programación una función que permite realizar esta tarea.

b) Filtrado de información no relevante. Los textos a tratar vienen a menudo acompañados

de otros elementos que deben ser eliminados o extraídos para facilitar el tratamiento. Así, si la fuente de información es una página de Internet, junto a los fragmentos de texto tratables, aparecen diferentes tipos de marcas que definen las características de visualización de la página, enlaces con otras páginas o dentro de la misma página, objetos no textuales, como por ejemplo imágenes, animaciones, tablas, gráficos, etc. Si el texto está marcado de forma consistente, el filtrado es relativamente sencillo, pero, a menudo no es así y la dificultad de la tarea aumenta. Otra tarea es la de la identificación de la lengua en los casos en que se desconozca *a priori* la lengua del texto, por ejemplo, si el texto ha sido recuperado a través de Internet, o en los casos en que un texto contenga fragmentos correspondientes a lenguas diferentes.

c) Localización de unidades tratables. Las unidades básicas de tratamiento son las palabras. Localizar las palabras ortográficas es sencillo si el espacio o los signos de puntuación actúan como separadores. En las lenguas en que esto no es así, por ejemplo, en el japonés, o en los casos en que la puntuación no existe, el problema es mayor. Pero aún en los casos en que se hayan localizado las palabras ortográficas existen casos problemáticos. A saber:

- Distinción entre palabras ortográficas y palabras gramaticales. Por ejemplo, en la conjunción 'sin embargo' en que una palabra gramatical corresponde a dos ortográficas o 'dímelo' en que una palabra ortográfica corresponde a tres palabras gramaticales, o contracciones como 'del' o 'al'.
- Términos multipalabra, como en el caso de 'San Pedro' o 'Buenos Aires'.
- Fechas, fórmulas, siglas, abreviaturas, etc.
- Nombres propios (de persona, geográficos, etc.).
- Palabras desconocidas, neologismos o errores. Es decir, palabras que no figuran en los diccionarios disponibles.

El uso diccionarios específicos o terminológicos que complementen los diccionarios generales, de procesadores para tratar las unidades no estándar (por ejemplo, extractores de fechas o identificadores de nombres propios), y de técnicas estadísticas y de aprendizaje automático contribuye a solucionar estos problemas.

1.4.2. Análisis morfológico: Etiquetado de partes del habla (*POS tagging*) y desambiguación de etiquetas

El siguiente paso en el tratamiento de la lengua consiste en el análisis morfológico. Esta tarea es normalmente realizada por un analizador morfológico cuyo papel es el de recuperar la

morfología de las palabras, es decir, las formas con que se construyen las palabras a partir de unidades significativas más pequeñas, llamadas 'morfemas'. Los morfemas se clasifican en dos clases: morfema raíz o lema (*stem*) y afijos. Generalmente, las palabras se forman a través de mecanismos de flexión, derivación o composición a partir de sus formas canónicas. La tarea de descomposición de una palabra de la entrada en su forma de base y sus afijos se denomina *stemming* o lematización.

Un analizador morfológico debe constar de por lo menos tres partes: un diccionario o lexicón con la lista de los lemas; una lista de afijos con sus reglas de orden, ya que los afijos no pueden aparecer en un orden arbitrario y un conjunto de reglas ortográficas en el caso que la adición de un afijo las requiera, como en el caso de *pez (sing) → peces (pl)*.

El análisis morfológico de las formas flexivas del español es relativamente sencillo, ya que la flexión responde a patrones bastante regulares. El número de sufijos flexivos es de unos 200 y reglas de combinación ascienden a unas 500. En cambio, la derivación o la composición son más complicadas y suelen venir combinadas con la flexión. A menudo, sobre todo, para lenguas con poca complejidad morfológica o para córpora pequeños el analizador morfológico se reduce a un diccionario de formas completas. Si el 'lexicón' está correctamente implementado, la eficiencia del proceso de análisis es alta. Por otra parte, los 'lexicones' son fácilmente extensibles, soportan entradas multipalabra y es posible su construcción a partir de generadores morfológicos.

Para que el procesamiento morfológico sea posible, cada lema debe ser previamente etiquetado. Se denomina 'etiquetado', *POS tagging* (del inglés, *part-of-speech tagging,* etiquetado de partes del habla) o simplemente *tagging* al procedimiento de asignar a cada una de las unidades léxicas presentes el conjunto de sus categorías gramaticales posibles (Jurafsky & Martin, 2000). El problema es que las palabras tomadas en forma aislada son ambiguas respecto de su categoría. Si se considera el siguiente ejemplo: 'Yo bajo con el hombre bajo a tocar el bajo bajo la escalera'. La palabra 'bajo' puede pertenecer, dependiendo del conjunto de etiquetas que se manejen, a un mínimo de cuatro categorías diferentes: verbo, adjetivo, nombre y preposición. El analizador morfológico devolverá toda ellas para cada una de las apariciones de la forma 'bajo' en la oración. Afortunadamente, la categoría de la mayoría de las palabras no es ambigua respecto de su contexto. Para un humano es relativamente simple eliminar la ambigüedad en la categorización, lo hace de manera rápida y eficiente, pero no lo es para una máquina. Para ello, existen los desambiguadores morfosintácticos (*POS taggers*) o etiquetadores, cuya misión es la de realizar automáticamente esta tarea.

El objetivo de un etiquetador es el de asignar a cada palabra la categoría más 'apropiada'

dentro de un contexto. Por supuesto, la calidad de éste dependerá del grado de precisión ('granularidad') del etiquetado, del contexto considerado y de la información de que se disponga para considerar *apropiada* una etiqueta o secuencia de etiquetas. A veces, los etiquetadores no resuelven totalmente el problema de la ambigüedad gramatical y se limitan a eliminar las opciones imposibles o menos probables. Este es el caso de los denominados 'desambiguadores reduccionistas'.

Existen tres grandes tipos de etiquetadores o métodos de etiquetado: los basados en reglas, los estadísticos o probabilísticas y los híbridos basados en transformaciones. Los 'etiquetadores basados en reglas' utilizan conocimiento lingüístico (*knowledge-driven taggers*), generalmente expresado en forma de reglas o restricciones para establecer las combinaciones de etiquetas aceptables o prohibidas. Las reglas se escriben manualmente, responden a criterios lingüísticos y se representan en forma explícita. Los primeros sistemas de etiquetado basado en reglas constaban de dos etapas. La primera etapa contenía un diccionario que asignaba a cada palabra una lista de todas las etiquetas posibles para esa palabra y la segunda etapa constaba de una lista de reglas de desambiguación escritas a mano para lograr que a cada palabra se le asigne una sola etiqueta. Se trata de sistemas de muy alta precisión, por ejemplo, el ENGTWOL de Karlsson (Voutilainen, 1995, cit. en Jurafsky & Martin, 2000) que implementa las denominadas *Constraint Grammars* (gramáticas de restricciones) para el inglés. Este sistema también está basado en una arquitectura de dos etapas, pero con un diccionario y reglas de desambiguación mucho más sofisticadas. El costo de desarrollo de este tipo de etiquetadores es alto y también lo es el costo de adaptación a otros dominios. Su precisión alcanza al 99.5%.

Los 'etiquetadores estadísticos' se basan en la evidencia empírica obtenida de corpus lingüísticos voluminosos (*data-driven taggers*). El costo es por ello mucho menor aunque también es menor su grado de precisión, superior en cualquier caso al 97%, suficiente en algunas aplicaciones. Los sistemas son independientes de la lengua y fácilmente adaptables a otras lenguas y dominios. El problema de estos sistemas reside en el aprendizaje del modelo estadístico utilizado. En este sentido es notable, y creciente, el uso de técnicas de aprendizaje automático. Se han utilizado técnicas de aprendizaje supervisado partiendo de corpus etiquetados manualmente y técnicas de aprendizaje no supervisado en las que no es precisa (o está limitada) esa intervención manual. Un algoritmo clásico utilizado para el etiquetado estadístico es el de los Modelos Ocultos de Markov (del inglés, *Hidden Markov Models,* HMM). Este enfoque se caracteriza por asumir que la probabilidad de una cadena de símbolos puede ser calculada en base a sus partes o 'n-gramas'. El modelo de 'n-gramas' más básico es el de los 'unigramas'; es decir, la búsqueda de la etiqueta más probable para cada palabra o *token.* Para ello, es necesario entrenar el sistema con un corpus etiquetado previamente.

Otros modelos algo más sofisticados de etiquetado son el de bi-gramas (*bigram-HMM tagger*) en los que la probabilidad de una etiqueta se estima simplemente con el contexto de la etiqueta anterior, o los tri-gramas en los que el contexto abarca a las dos etiquetas precedentes. En resumen, un modelo de 'n-gramas' considerará la palabra actual y la etiqueta anterior para determinar la etiqueta más probable para esa palabra. En la actualidad, los etiquetadotes HMM no solo determinan la etiqueta más probable para una palabra sino para toda una secuencia de palabras u oración, para lo cual se utiliza el algoritmo de Viterbi que calcula la trayectoria más probable en un HMM (Jurafsky & Martin, 2000; Siddiqui & Tiwary, 2008).

Por ejemplo, si se considera la siguiente oración:

El ave puede volar

con su secuencia de etiquetas:

DT (determinante) MD (verbo modal) NN (nombre) VB (verbo) utilizando el sistema de bigramas:

```
          DT  NN  MD        VB
P <       |         |       |         |
          El  ave   puede   volar
```

La probabilidad se puede calcular de la siguiente manera:

= P (DT) x P (NN|DT) * P (MD|NN) x P (VB|MD)

x P (*el*/DT) x (*ave*/NN) x (*puede*/MD) x P(*volar*/VB)

En tercer lugar, los 'sistemas híbridos' combinan métodos estadísticos y basados en reglas para intentar recoger los aspectos positivos de cada una de ellas y evitar sus limitaciones. Un ejemplo de este sistema son los sistemas basados en transformaciones (del inglés, *Transformation-Based-Tagging*, también llamado *Transformation-Based-Learning (*TBL*)* o *Brill Tagging)*. Este sistema fue introducido por Brill (1995) y se basa en el aprendizaje automático (Jurafsky & Martin, 2000). Cada palabra se rotula con la etiqueta más probable, luego se cambia la etiqueta aplicando reglas del tipo 'si palabra -1 es un determinante cambie la etiqueta a nombre' y se re-etiqueta la palabra. Se obtiene de esta manera una secuencia de reglas de transformación. Recientemente, han comenzado a utilizarse sistemas de desambiguación por combinación. Se trata de combinación de diferentes modelos del lenguaje en un único desambiguador, de combinación de desambiguadores mediante votación u otros procedimientos más sofisticados de aprendizaje (Jurafsky & Martin, 2000; Siddiqui & Tiwary, 2008).

Yo	les	hablaría	de	usted	.
yo	*les*	*hablar*	*de*	*tú*	.
PP1CSN00	PP3CPD00	VMIC1S0	SPS00	PP2CS00P	Fp

Figura 1. Ejemplo de procesamiento morfológico en *Free Ling 2.1*.

La Figura 1 muestra un ejemplo de procesamiento morfológico en *Free Ling 2.1* con lematización y etiquetado de partes del habla.

1.4.3. Análisis sintáctico o *parsing*

Una vez analizado morfológicamente y etiquetado o desambiguado total o parcialmente el texto, puede realizarse el análisis sintáctico.

El *parsing* es un proceso por medio del cual se convierte el texto de entrada en otras estructuras (comúnmente árboles), que son más útiles para el posterior análisis y capturan la jerarquía implícita de la entrada. Durante el procesamiento se producen distintas estructuras intermedias o de trabajo, hasta producir un árbol de análisis estructural de la secuencia de entrada. Un árbol correcto es aquél que cubre todos y solo los elementos del enunciado y en cuyo tope tiene un símbolo S (del inglés *sentence)* u O (de 'oración') (Lavid, 2005; Jurafsky & Martin, 2000).

Hay diferentes técnicas y algoritmos de *parsing*. Estas se pueden agrupar básicamente entre tres tipos diferentes:

a) Procesamiento paralelo o secuencial de las alternativas. Se refiere fundamentalmente a dos tipos de análisis de secuencias. La técnica de procesamiento 'en paralelo' prueba diferentes posibilidades de combinación en paralelo y guarda la pista de los estados simultáneos posibles. La estrategia de procesamiento 'secuencial o en profundidad' prueba primero una posibilidad hasta el final, y si no tiene éxito, retrocede al punto de partida y prueba otra ruta hasta dar con la estructura que corresponde a la secuencia de la entrada.

b) Procesamiento descendente o ascendente. Se refiere al punto de partida del árbol estructural que el *parser* debe construir. Si se está procesando una oración, en la parte superior se encontrará un símbolo inicial (O) que representa a la oración en su totalidad y, en la parte inferior, del árbol hay nodos que representan los elementos léxicos individuales, las palabras. La dirección ascendente y descendente dependen del punto de partida: si comienza el procesamiento en la parte superior con el símbolo inicial (O) y va dividiendo la entrada progresivamen-

te en partes cada vez más pequeñas, hasta llegar a las palabras, será un *parser* descendente (*top-down-parser)*. El *parser* será ascendente *(bottom-up),* si por el contrario, el análisis comienza por los elementos léxicos individuales y culmina con el símbolo inicial (O).

c) Procesamiento determinista/no determinista. Se refiere al carácter guiado o no guiado del modelo. Es decir, si el modelo no permite decidir qué regla de la gramática se aplicará en un momento determinado, se tratará de un modelo 'no determinista'; en cambio, si se utilizan mecanismos que conducen a un resultado concreto sin vacilaciones, se hablará de un 'procesamiento determinista'. Además de estas, existen muchas técnicas de *parsing*. Sin embargo, muchas veces persisten los dos grandes problemas propios de estos analizadores, la ambigüedad y el costo informático que implica el tiempo de procesamiento.

Se pueden solucionar los problemas de ambigüedad y pensar en un análisis sintáctico en profundidad. Sin embargo, hay ocasiones en que este análisis no es posible o no es conveniente porque el procesamiento es muy lento y/o costoso. En este caso, se puede realizar un análisis superficial o fragmental en lugar de un análisis en profundidad. Para muchas aplicaciones no es necesario desarrollar todo el análisis.

Se ha señalado la insuficiencia de los métodos convencionales de análisis sintáctico para tratar textos no restringidos. Problemas como la dificultad de una segmentación adecuada, la obtención de no uno sino varios (a menudo muchos) árboles de análisis o la necesidad de ampliar la cobertura del analizador al tratamiento de oraciones no gramaticales o que incluyan palabras desconocidas tienen difícil solución en el marco tradicional de un analizador sintáctico que trate de obtener un árbol de análisis completo del texto basándose en una gramática de amplia cobertura. Ante ello caben dos aproximaciones: analizar en profundidad pero no todo, caso del 'análisis fragmental', y analizar todo pero no en profundidad, caso del 'análisis superficial'.

El objetivo de los analizadores fragmentales, también denominados agrupadores sintácticos o *chunkers*, es la detección de frases nominales, verbales, adjetivas, adverbiales básicas (sin recursión). A veces se trata simplemente de detectar el segmento (es lo que se denomina 'parentizado' o *bracketting*), mientras que en otras ocasiones se desea obtener el etiquetado correcto y la estructura sintáctica del segmento. Es frecuente el uso de técnicas de estados finitos y la actuación de transductores en cascada. También se han utilizado técnicas de aprendizaje automático. A menudo el siguiente paso es la obtención de dependencias y de relaciones sintácticas entre los segmentos (Rodríguez, 2000). Otro tipo de *parser* es el *parser* superficial o parcial se denomina *shallow parsing* o análisis ligero. Este se encarga solo de encontrar los componentes principales de la frase como nombres o verbos.

La mayoría de los analizadores modernos son al menos en parte estadísticos, esto quiere decir

que se basan en unos datos de entrenamiento que han sido analizados manualmente. Este enfoque permite al sistema reunir información sobre la frecuencia con que ocurren ciertas construcciones en un contexto específico. Algunos de estos enfoques han incluido gramáticas libres de contexto probabilísticas, sistemas de máxima entropía (es decir, métodos de aprendizaje probabilísticos basados en corpus) y redes neuronales.

Los *parsers* diseñados para la enseñanza de lenguas generalmente contienen un componente que anticipa o detecta los errores en el caso de que las reglas gramaticales sean violadas. Por ejemplo, un procedimiento muy común son las *buggy rules* (reglas agramaticales), que hacen posible que el *parser* procese una oración que contiene uno o más errores e identificarlos. Investigadores del área (Heift & Schulze, 2007) analizaron, con la ayuda de métodos estadísticos, córpora de alumnos de L2 para poder determinar las *buggy rules* necesarias para ser incluidas en el *parser*. La importancia de los *parsers* en CALL fue muy discutida en la última década por varios investigadores: Nagata, Matthews, Holland, Maisano, entre otros (Heift & Schulze, 2007; Schulze, 2008). Holland et al. (1995) se refieren a las posibilidades y limitaciones de tutores de lengua basados en parser, realizan una comparación entre el CALL convencional y el CALL basado en *parser* y concluyen que en CALL basado en *parser* el estudiante puede escribir una gran variedad de oraciones y desarrollar de una forma relativamente libre habilidades para el mejoramiento de la producción escrita.

El componente de *parse*r consta de un pre-procesador de texto, un analizador morfológico, un diccionario y un *parser* que proporciona el árbol sintáctico.

Tal como se observa en la Figura 2, la entrada del alumno/usuario '*La secretaria han escrito varias cartas' es analizada en primer lugar por el preprocesador de texto que analiza y subsana posibles errores de escritura comparándola con el lexicón; luego es procesada por el analizador morfológico que compara esta frase con la gramática del sistema y realiza el análisis sintáctico proporcionando un árbol; en esta etapa el *parser* detecta un error de concordancia y se genera el *feedback*. 'El sujeto 'secretaria' no está en concordancia con el verbo han'. De esta forma, la salida del sistema con la detección del error permitirá en otra etapa del Sistema Tutorial Inteligente una estrategia de *feedback ad hoc* al tipo de error detectado por el *parser.*

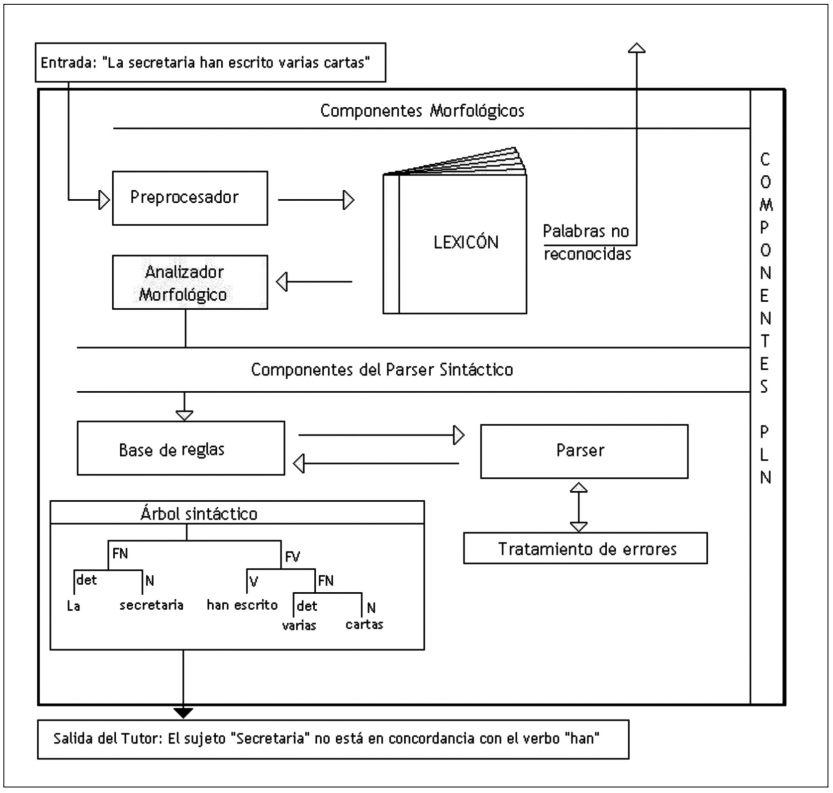

Figura 2. Diagrama simplificado de un componente de *parsing*.

1.5. Los Sistemas Tutoriales Inteligentes y el aprendizaje de lenguas extranjeras

Como ya se explicitó en la introducción, los STI para LE han incorporado técnicas de NLP, por ejemplo, para el análisis de las entradas en lenguaje natural de los estudiantes o para modelar la competencia de una lengua extranjera. Esto es, para que los alumnos /usuarios del sistema tengan una retroalimentación más precisa y flexible. Estos sistemas utilizan técnicas específicas de analizadores sintácticos (*parsing*) para analizar las entradas de los alumnos e identificar los errores que cometen en dichas oraciones. Estas técnicas de NLP han permitido a los sistemas manejar algunas estrategias de *feedback* más sofisticadas como las claves metalingüísticas

y los 'informes de errores' basados en los análisis de los errores (Ferreira, Moore & Mellish, 2007). Sin embargo, algunos STI parecen haber sido creados sin referencia a los estudios que existen en relación con las habilidades de los estudiantes de lenguas (Holland et al., 1995; Bull, 1997; Chapelle, 1997). Para mejorar la efectividad de dichos sistemas, los investigadores deben poner mayor atención en los resultados que emergen en la literatura especializada de Adquisición de Segundas Lenguas. No está claro que se pueda formalizar principios tutoriales que sirvan como estructuras de control de un sistema tutorial inteligente sin haber examinado antes las interacciones presenciales en la modalidad tradicional (*face-to-face*) y tutorial (*one-to-one*) (Tomlin, 1995). Es así como muchos estudios empíricos previos en el área de los STI procedimentales han utilizado como modelos las interacciones de un estudiante y su tutor (modalidad *one-to-one*, presencial). Sin embargo, en la enseñanza de lenguas extranjeras, la interacción usualmente toma lugar en ambientes de clases tradicionales (modalidad *face-to-face*, presencial). Por esa razón, hemos llevado a cabo estudios empíricos previos (Ferreira, 2003, 2006, 2007) con el objeto de investigar el tipo, frecuencia y efectividad de las estrategias de feedback utilizadas en la enseñanza del español como lengua extranjera en tres contextos de aprendizaje de lengua distintos: un estudio observacional de interacciones profesor-alumno en clases tradicionales presenciales (modalidad *face-to-face*), un estudio de caso de interacciones en clases tutoriales presenciales (modalidad *one-to-one*), y un estudio experimental en el cual los estudiantes interactúan con una aplicación computacional 'no presencial' (modalidad *one-to-one* a distancia) accesada en línea a través de Internet. Los resultados de estos tres estudios empíricos han mostrado ser similares en relación con los tipos, frecuencia, y efectividad de las distintas estrategias de *feedback* correctivo que se investigaron. Se propone, entonces, que los STI para lenguas extranjeras deberían implementar estrategias de *feedback* correctivo que estimulen a los estudiantes a la auto-reparación de sus errores. Hemos definido un modelo para el diseño de un componente de estrategias de *feedback* de un sistema tutorial inteligente para el español como lengua extranjera (Figura 2) que toma en cuenta: el tipo de error que el estudiante ha cometido (gramática, vocabulario o pronunciación) y el nivel de aprendizaje del estudiante (principiante, intermedio y avanzado). En nuestro modelo, se asume que el análisis del error es llevado a cabo por un módulo 'analizador/intérprete'. Como se señaló en el marco teórico, sistemas tutoriales inteligentes para lenguas extranjeras han hecho uso exitoso de la tecnología de *parsing* (analizadores) para identificar errores de gramática. Por otra parte, los avances en la investigación en sistemas de diálogo hacen que las técnicas más interactivas que se requieren para incorporar este tipo de estrategias sean más factibles de implementar en un STI para LE (Owen, Schultz, Peters, Chen & Pon-Barry, 2005; Zinn, Moore & Core, 2005).

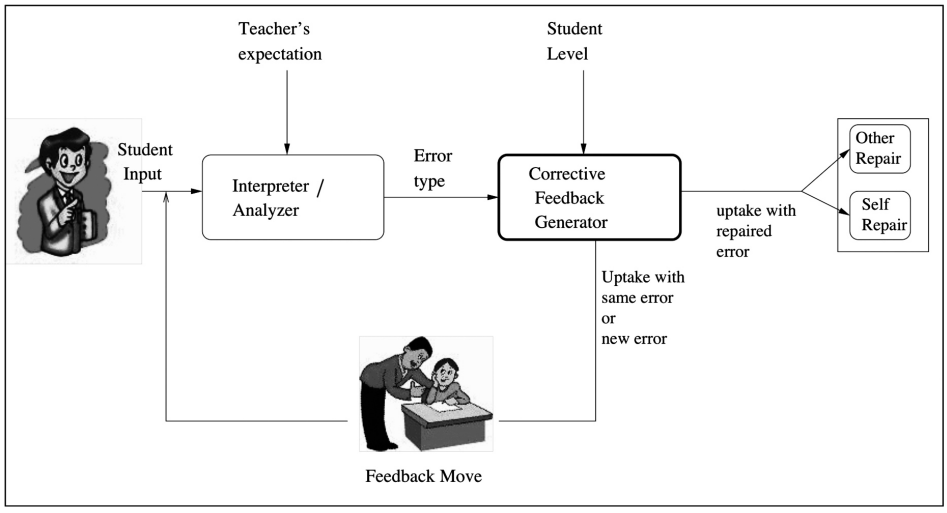

Figura 3. Modelo del Proceso del Tratamiento del Error y la Generación de *Feedback* para un Sistema Tutorial Inteligente para lenguas extranjeras.

En nuestro modelo (Figura 3), la secuencia comienza con una respuesta del estudiante que contiene al menos un error. En el caso de que aparezca más de un error, es necesario tomar una decisión acerca de qué error debería ser tratado primero. Esto se determina de acuerdo al nivel de aprendizaje del estudiante. Para estudiantes principiantes, los errores de gramática y pronunciación resultaron ser más frecuentes en nuestros dos primeros estudios. En consecuencia, se sugiere que prioritariamente deberían ser tratados por el sistema estos dos tipos de errores. Para estudiantes de nivel de aprendizaje intermedio o avanzado, los errores de gramática y vocabulario deberían ser tratados primeramente.

1.5.1. El análisis de los errores

Los errores y el *feedback* correctivo son parte natural del aprendizaje de la lengua. Los errores pueden definirse como desviaciones de las normas de la lengua meta (Ellis, 1997). Los errores revelan patrones de desarrollo de los sistemas de interlengua de los estudiantes que aprenden una segunda lengua, señalando dónde ellos sobregeneralizan una regla o dónde transfieren de manera inapropiada una regla de la lengua materna a la segunda lengua (Lightbown & Spada, 1990, 1999). El *feedback* correctivo se refiere a una indicación para el estudiante de que alguna

estructura en la lengua meta está incorrecta. De acuerdo con Ellis (1997), el principal hallazgo de los estudios sobre el tratamiento de errores es que éste es un proceso enormemente complejo. Esto puede observarse en las diferentes taxonomías de *feedback* propuestas (Long, 1977; Day, 1984; Chaudron, 1977; Lyster, 1997; Seedhouse, 1997). Algunas estrategias de *feedback* propuestas son marcadamente diferentes de aquéllas que típicamente se implementan en STI de dominios procedimentales.

Dado que el *parser* de un STI para lenguas extranjeras debe ser capaz de procesar entradas erróneas, las teorías tradicionales de análisis de errores han recobrado vigencia y una nueva perspectiva. "El análisis de errores es el proceso que determina la incidencia, la naturaleza, las causas y las consecuencias de los fracasos lingüísticos" (James, 1998: 1). El tratamiento de errores gramaticales ha recibido máxima atención en CALL basado en *parser* o ICALL. La identificación y la descripción de errores juegan un rol importante en la investigación, desarrollo y posterior diseño de *parsers* que pueden procesar entradas erróneas.

La teoría del Análisis Contrastivo (CA) (Lado, 1957) trataba de explicar todos los errores de los alumnos a través de la comparación con su lengua nativa. Este proceso es descrito como transferencia o interferencia. Los seguidores del CA argumentan que la lengua del alumno puede ser 'construida' contrastando el sistema de la L1 con el de la L2. Según sus postulados los fenómenos que más difieren entre ambas lenguas son los más difíciles de adquirir y tienen mayor probabilidad de errores (i.e. la flexión de los adjetivos en una lengua y no en la otra). Sin embargo, en aquellos fenómenos que no difieren mucho (i.e. algunas letras tienen el mismo sonido en ambas lenguas), son más fáciles de adquirir y con menos probabilidades de cometer errores. Sin embargo, se ha comprobado que muchos de los errores que cometen los alumnos no son atribuibles a la interferencia con la lengua materna y, más bien, provienen de errores en la aplicación de las reglas de la lengua meta.

Un importante prerrequisito para la adecuada descripción de los errores y la generación de respuestas útiles en CALL es una apropiada clasificación de los errores. El uso de una clasificación de errores tiene las siguientes ventajas:

- La descripción de errores y sus respuestas pueden estar basadas tanto en errores individuales como en clases de errores.
- Los errores que pertenecen a una misma clase pueden ser tratados de manera similar.
- La clasificación de los errores permite a los sistemas basados en *parser* no solo guardar los errores, sino también calcular el número errores que pertenecen a una misma clase. Lo que resulta en un *feedback* más efectivo, ya que puede estar basado en un modelo de alumno más claro y comprensible.

2. Diseño e implementación de un analizador para el Español como LE

La particularidad de este analizador gramatical de entradas (*parser*) es que debe procesar entradas erróneas y para ello, es necesario poder predecir los errores que puede cometer el usuario en un momento determinado de su aprendizaje y en un tópico gramatical específico. Hoy en día no existe una tipología de errores universal, disponible para cualquier sistema lingüístico que se desee tratar. Esto se debe a que todas las lenguas son diferentes y los tipos de errores identificados pueden variar considerablemente. Así pues, una tipología de errores no puede ser elaborada independientemente del objetivo para el cual ha sido concebida. El objetivo en particular es contrastar la taxonomía de base teórica basada en la literatura especializada con las muestras empíricas resultantes de un estudio observacional en clases tradicionales con el fin de obtener una información más acotada de los errores que podrían llegar a cometer alumnos de ELE con lengua materna inglés en su mayoría.

Este análisis da como resultado una taxonomía de errores acotada a la realidad muestral, pero que debe ser nuevamente ajustada para ser aplicada al tutorial con características particulares y con un modelo de alumno específico. Para ello es necesario trabajar además en la selección de ejercicios que sean adecuados para un tutor inteligente. Tenemos que tener en cuenta que el tutor debe 'comprender' de manera automática la frase entregada por el usuario, debe realizar el análisis, extraer los errores si los hay y dar una respuesta acorde.

- Elaboración de ejercicios.

Se diseñó una serie de ejercicios teniendo en cuenta el nivel de proficiencia del alumno. Para ello, se operó con los descriptores del ALTE para el nivel de experticia con que funcionará el tutor, en este caso, el nivel B2. También se tuvo en cuenta, no solo las respuestas correctas esperables según el ejercicio en cuestión, sino que también las posibles respuestas incorrectas que tendrá que detectar el analizador, tomando como base la taxonomía de errores.

- Etiquetado de partes del habla o *POS Tagging*.

Para el correcto funcionamiento del analizador sintáctico debe realizarse previamente o de manera simultánea un análisis morfológico. Los analizadores morfológicos para el español utilizan una serie de etiquetas para representar la información morfológica de las palabras. Este conjunto de etiquetas se basa en las etiquetas por el grupo EAGLES para la anotación morfosintáctica de lexicones y corpus para las lenguas europeas. Esto es independiente de que haya atributos que no estén especificados y, por lo tanto, haya que crear etiquetas *ad hoc* o, por el contrario, etiquetas que para el corpus no se consideran relevantes y no haya que incluirlas.

- Elaboración de la gramática.

Para el funcionamiento del *parser,* es necesario construir una gramática, es decir, un conjunto de reglas de formación de frases que el analizador reconocerá. Además, es necesario definir un conjunto de reglas agramaticales (*buggy rules)*, lo que permite que el *parser* procese una oración que contiene uno o más errores e identificarlos.

- Jerarquización de errores.

Debido a que el *parser* debe procesar entradas erróneas, un elemento importante es el tratamiento de los errores. Este debe tener especificaciones muy precisas, debe tolerar, detectar y diagnosticar errores, como asimismo ser capaz de entregar una respuesta flexible en el reporte de los mismos. Un tema importante es el tratamiento de los errores múltiples. Por un lado, es deseable que el programa sea capaz de detectar y explicar todos los errores, pero eso no significa que tenga que desplegar cada uno de los errores detectados. La ausencia de un mecanismo de filtro puede llegar a agobiar al alumno. Por ello, nuestro sistema contiene una jerarquía de errores, que ejemplificaremos con el caso de los ejercicios de futuro simple y compuesto:

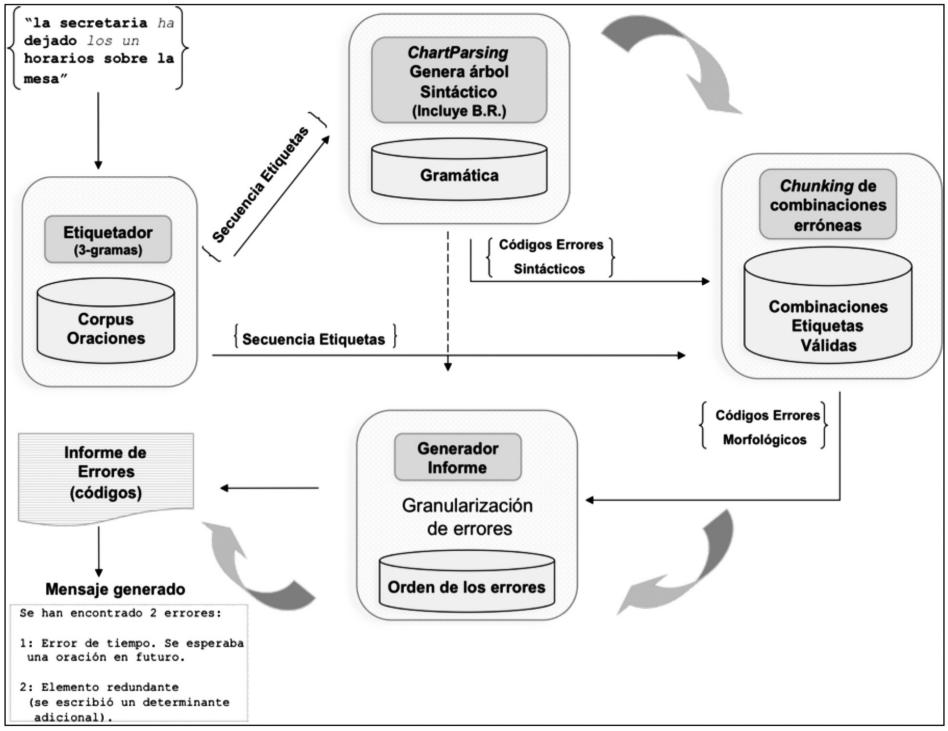

Figura 4. Modelo del *parser* ELE-UDEC del Sistema Tutorial Inteligente para el Español.

En la Figura 4 se precisan las distintas etapas del procesamiento lingüístico de la entrada "*la secretaria ha dejado los un horarios sobre la mesa" realizada por el *parser* ELE-UDEC. Esto es:

Procesamiento entrada

1. El estudiante ingresa una respuesta al ejercicio presentado por el Sistema Tutorial Inteligente.
2. Se etiqueta la entrada usando un etiquetador que aplica la técnica de 'n-gramas' (en este *parser* en particular se utiliza un sistema de 3-gramas que consiste en la revisión del contexto de dos etiquetas precedentes).

3. La oración etiquetada es utilizada para el análisis sintáctico y morfológico.

Análisis morfológico y sintáctico

4. El análisis morfológico utiliza una técnica llamada *chunking,* que sirve para la detección de ciertas identidades o secuencias específicas de texto, de esta forma esta técnica se usa para buscar errores morfológicos dentro de una oración mediante reglas similares a las *buggy rules,* pero que en este caso se centra en la búsqueda de secuencias de etiquetas específicas dentro de una oración que representa un error morfológico, como por ejemplo: una oración con error de tiempo o número. Los errores morfológicos son informados mediante una codificación del mismo. Actualmente se informa el tipo de error (modo, número, género, etc.), seguido del rasgo esperado, por ejemplo, 'error de número, se esperaba una oración escrita en singular'.
5. El análisis sintáctico utiliza una técnica de *parsing* basado en una gramática de contexto libre o de estructura de frase. En particular, se utiliza *chart-parsing*, que se diferencia de los otros métodos de *parsing* por la utilización de la programación dinámica, lo que la hace más eficiente en lenguas ambiguas, además evita el *back-tracking* o retroceso y previene de una explosión combinatoria. Esta gramática incluye reglas de errores sintácticos que corresponden principalmente a la estructura de la oración (por ejemplo, errores de orden, de omisión y adición de palabras entre otros, mediante *buggy rules*) y de esta forma los errores son detectados mediante la inspección del árbol generado por el *parsing*.
6. Una vez que se tienen los errores tanto sintácticos como morfológicos, se desarrolla un informe que da cuenta de los errores encontrados de acuerdo a la granularidad de errores definida.

3. Salidas del *parser* ELE-UDEC

A continuación se muestran algunos ejemplos de los análisis sintácticos realizados por el *parser* ELE-UDEC. La Figura 5 muestra dos entradas del alumno en un ejercicio de futuro; la primera está correcta, lo que es señalado en color verde; la segunda respuesta es incorrecta y esto se señala en color rojo y con el *feedback:* 'se ha encontrado un error' y 'la respuesta no es una oración, no hay verbo'. Con esto se le indica al alumno que toda oración del español debe contener por lo menos un verbo conjugado.

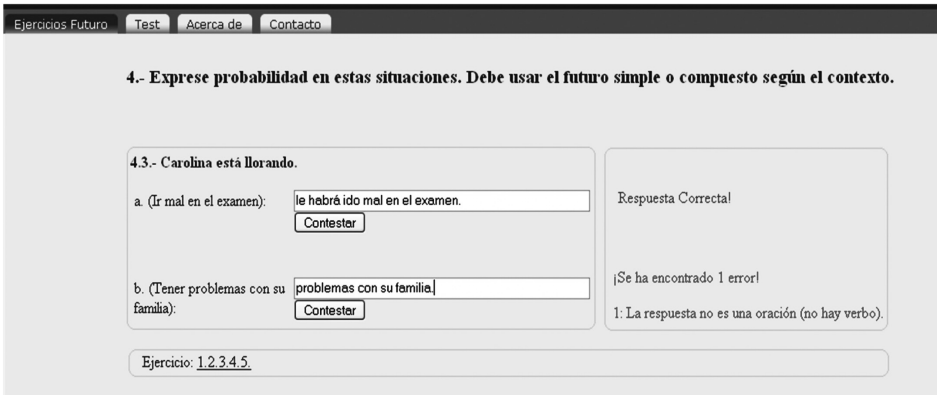

Figura 5. Identificación de la respuesta correcta y del tipo de error 'omisión del verbo' realizada por el *parser* ELE-UDEC.

La Figura 6 muestra una particularidad de este *parser*; este no solo debe identificar los errores en la forma gramatical, sino que debe tener en cuenta el tipo de ejercicio que debe desarrollar el alumno; es decir, ser coherente discursivamente con lo planteado en el ejercicio. Por ejemplo, si el alumno debe responder con una oración de futuro y responde en pasado, aun cuando la oración esté gramaticalmente correcta, el sistema debe ser capaz de informar al alumno que no está cumpliendo con los requerimientos del ejercicio.

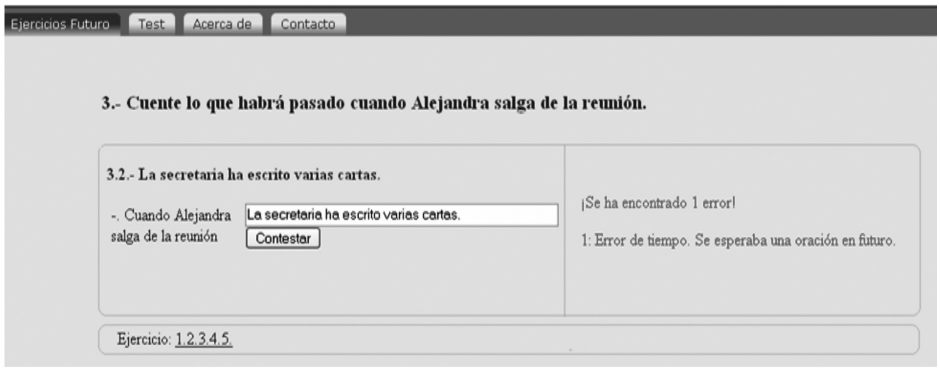

Figura 6. Identificación del tipo de 'error en el tiempo verbal respecto de la expectativa del ejercicio' realizada por el *parser* ELE-UDEC.

La Figura 7 ilustra el problema de los errores múltiples en una misma oración o frase; en efecto, el alumno puede presentar más de un error en una misma frase; es por eso, que el *parser*, de acuerdo con una jerarquización de errores con que fue programado, despliega los errores por orden de importancia.

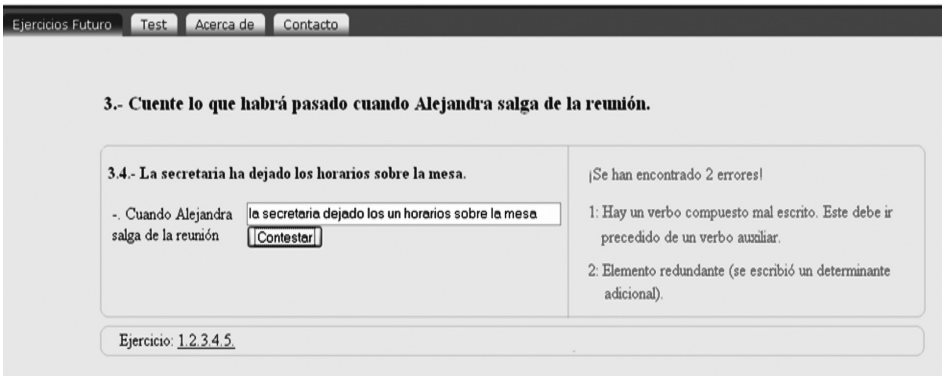

Figura 7. Identificación de errores múltiples realizada por el *parser* ELE-UDEC.

En el caso de que los errores sean de la misma importancia según la jerarquización de errores del sistema, el *parser* desplegará los errores de manera aleatoria. Lo que es señalado en la Figura 8.

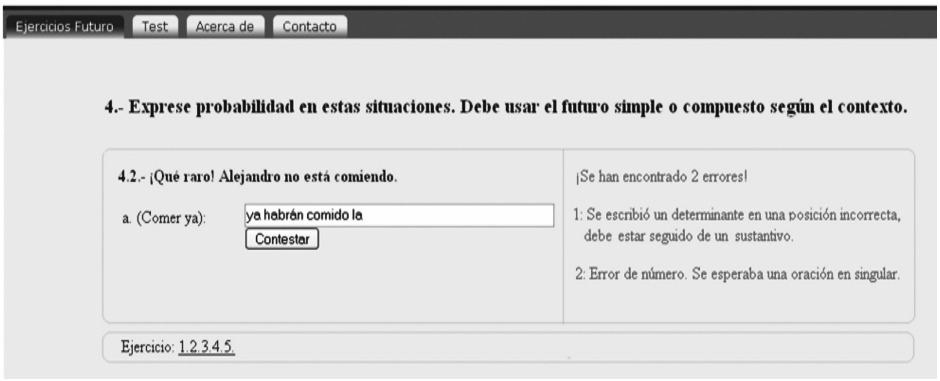

Figura 8. Identificación de errores múltiples realizada por el *parser* ELE-UDEC.

Comentarios finales

Este artículo ha centrado su atención en el diseño y la implementación de un analizador automático de errores gramaticales para el español en el contexto de un tutorial inteligente para la enseñanza del español como lengua extranjera: *Parser ELE-UDEC*. En general, la fundamentación teórica, empírica y ejemplos expuestos han evidenciado que el diseño e implementación propuestos en este enfoque están conduciendo a la identificación y reconocimiento de los tipos de errores cubiertos en este dominio restringido de la enseñanza del español cono lengua extranjera apoyando de esta forma el proceso de enseñanza aprendizaje en lo que respecta a la precisión de las formas lingüísticas gramaticales que se requieren desarrollar en la competencia lingüística de los estudiantes de B2.

Un tema pendiente que debería ser considerado en la implementación de nuestro Sistema Tutorial Inteligente es cómo las estrategias de *feedback* deberían ser diseñadas y generadas para el procesamiento de lenguaje natural (PLN). Por otra parte, con el fin de que estas estrategias de *feedback* sean efectivas como componente de un STI para una lengua extranjera, otros temas claves que deberían abordarse están relacionados con el modelo del tutor y el modelo del estudiante. El tratamiento de este tipo de estrategias *feedback* correctivo en el contexto de un modelo del tutor requiere considerar aspectos tales como:

- La incorporación de estrategias de *feedback* de manera natural y auténtica dentro de un enfoque de enseñanza.
- La definición del grado de explicitación de las estrategias de *feedback*. Es necesario elegir entre estrategias de *feedback* que llaman la atención del estudiante hacia el error de manera no obstructiva y aquellas que dirigen la atención del estudiante al área del problema de manera más explícita.
- Tomar en cuenta el grado de efectividad de las estrategias de *feedback* de acuerdo con los tipos de error, nivel de aprendizaje y tipo de estrategia.
- Determinar tipos de errores acordes al nivel de proficiencia que se seleccionarán en el contexto de un proceso de tratamiento dirigido a formas de la lengua que se constituyen en excelentes candidatas para ser tratadas en cualquier momento del proceso de enseñanza-aprendizaje (ej: tiempos, concordancia, etc.).
- Delimitar las dificultades de aprendizaje a medida que aparecen.

En tanto, para un Modelo del Estudiante, el tratamiento efectivo de los errores dentro de este enfoque requiere que se considere, entre otros, los siguientes aspectos:

- Conocimiento de los errores del estudiante; es decir, los registros y control de errores en las diferentes interacciones donde se involucra el estudiante.
- La especificación de la etapa del proceso de aprendizaje del estudiante para que se beneficie de la corrección.
- La percepción del estudiante acerca de las estrategias de *feedback*: los estudiantes deben ser capaces de darse cuenta de la forma correcta proporcionada por el tutor a través de la estrategia de *feedback*.
- La actitud del estudiante ante el *feedback* (reparación de otro o auto-reparación).

Finalmente, nuestro enfoque de investigación ha sido enriquecido por la investigación de diferentes disciplinas, incluyendo la adquisición de segundas lenguas (ASL), los sistemas tutoriales inteligentes en contextos procedimentales, y los sistemas tutoriales inteligentes para el área de lenguas extranjeras. Estas diversas perspectivas conducen a responder una pregunta general acerca de cómo los STI para LE pueden contribuir a aliviar las limitaciones o desventajas observadas en la modalidad presencial de clases tradicionales ya sea proveyendo mayor oportunidad de interacción a los estudiantes durante la clase o bien tratando los errores de los estudiantes con adecuadas y efectivas estrategias de *feedback* correctivo, como aquellas que apelan a que ellos sean capaces de resolver por sí mismos sus errores de lengua. De esta forma, este tipo de investigación no solo favorece un área específica de estudio sino que nutre tanto a las modalidades presenciales como no presenciales y semi-presenciales de enseñanza de lenguas.

REFERENCIAS BIBLIOGRÁFICAS

Bull, S. (1997). Promoting effective learning strategy use in CALL. *Computer Assisted Language Learning*, *10*(1)3-39.

Chapelle, C. A. (1997). CALL in the year 2000: Still in search of research paradigms? *Language Learning and Technology,* 2(1), 22-34.

Chaudron, C. (1977). A descriptive model of discourse in the corrective treatment of learner`s errors. *Language Learning*, 27, 29-46.

Corder, P. (1981). *Error analysis and interlanguage*. Oxford: Oxford University Press.

Day, R. (1984). Students participation in the ESL classroom, or some imperfections of practice. *Language Learning*, 34, 69-102.

Dodigovic, M. (2005). *Artificial intelligence in second language learning: Raising error awareness*. Clevedon: Multilingual Matters Ltd.

Ellis, R. (1997). *The study of second language acquisition*. Oxford: University Press.

Ferreira, A. (2003). *Feedback strategies for second language teaching with implications for Intelligent Tutorial Systems*. Ph.D. Thesis, University of Edinburgh, Edinburgh, UK.

Ferreira, A. (2006). Estrategias efectivas de Feedback positivo y correctivo en español como lengua extranjera. *Revista Signos, 39*(62), 309-406.

Ferreira, A. (2007). Estrategias efectivas de feedback correctivo para el aprendizaje de lenguas asistido por computadores. *Revista Signos, 40*(65), 521-544.

Ferreira, A., Moore, J. & Mellish, C. (2007). A study of feedback strategies in foreign language classrooms and tutorials with implications for Intelligent computer-assisted language learning systems. *International Journal of Artificial Intelligence in Education, 17*(4), 389-422.

Free Ling 2.1. [en línea]. Disponible en: http://garraf.epsevg.upc.es/freeling/demo.php

Graesser, A. C., Jeon, M., Yang, Y. & Cai, Z. (2007). Discourse cohesion in text and tutorial dialogue. *Information Design Journal, 15*(3), 199-213.

Graesser, A.C., Person, N., Harter, D. & TRG (2001). Teaching tactics and dialog in AutoTutor. *International Journal of Artificial Intelligence in Education*. Versión en PDF.

Heift, T. (2003). Multiple learner errors and feedback: A challenge for ICALL systems. *CALICO Journal, 20*(3), 549-560.

Heift, T. & Schulze, M. (2007). *Errors and intelligence in computer-assisted language learning. Parsers and Pedagogues*. Nueva York: Routledge.

Holland, M., Kaplan, J. & Sams, M. (1995). *Intelligent language tutors: Theory shaping technology*. Nueva Jersey: Lawrence Erlbaum Associates. U.S. Army Research Institute.

James, C. (1998). *Error in language learning and use. Exploring error analysis*. Harlow: Pearson Education Limited.

Jurafsky, D. & Martin, J. (2000). *Speech and language processing. An introduction to natural language processing, Computational linguistics, and speech recognition*. Nueva Jersey: Prentice Hall Inc.

Lado, R. (1957). *Linguistics across cultures, applied linguistics language leachers*. Ann Arbor, MI: University of Michigan Press.

Lavid, J. (2005). *Lenguaje y nuevas tecnologías. Nuevas perspectivas, métodos y herramientas para el lingüista del siglo XXI*. Madrid: Cátedra.

Lightbown, P. & Spada, N. (1990). Focus-on form and corrective feedback in communicative language teaching: Effects on second language learning. *Second Language Acquisitio*n, 12, 429-448.

Lightbown, P. & Spada, N. (1999). *How languages are learned*. Oxford: University Press

Long, M. (1977). Teacher feedback on learner error: mapping cognitions. En H. Brown, C. Yorio & R. Crymes (Eds.), *TESOL`77* (pp. 278–295). *Washington D.C.: TESOL*

Lyster, R. (1997). Recasts, repetition, and ambiguity in L2 classroom discourse. *Studies in Second Language Acquisition*, 20, 51-81.

Nagata, N. (1995). A study of consciousness-raising by computer: The effect of metalinguistic feedback on second language learning. *Foreign Language Annals*, *28*(3), 337–347.

Nagata, N. (1997). The effectiveness of computer-assisted metalinguistic instruction: A case study in Japanese. *Foreign Language Annals, 30*(2), 187–199.

Owen, E., Schultz, K., Peters, S., Chen, T. & Pon-Barry, H. (2005). Empirical foundations for intelligent coaching systems. *Proceedings of the interservice/industry Training, Simulation and Education Conference*, Orlando, Florida.

Rodríguez, H. (2000). Técnicas básicas en el tratamiento informático de la lengua [en línea]. Disponible en: http://dialnet.unirioja.es/servlet/oaiart?codigo=818617

Schulze, M. (2008). AI in CALL: Artificially Inflated or Almost Imminent? *Calico Journal, 25*(3), 510-527.

Seedhouse, P. (1997). The case of missing 'no': The relationship between pedagogy and interaction. *Language Learning*, 47, 547-583.

Siddiqui, T. & Tiwary, U. S. (2008). *Natural language processing and information retrieval*. Nueva Dehli: Oxford University Press.

Toole, J. & Heift, T. (2002). Task generator: A portable system for generating learning tasks for intelligent language tutoring systems. En P. Barger & S. Rebelsky (Eds.), *Proceedings of ED-MEDIA 02, World Conference of on Education Multimedia, Hypermedia* & *Telecomunicatios,* Charlottesville, VA: AACE.

Tomlin. R. (1995). Modeling individual tutorial interactions: Theoretical and empirical bases of ICALL. En M. Holland, J. Kaplan & M. Sams (Eds.), *Intelligent Language Tutors: Theory Shaping Technology* (pp. 221-241). Nueva Jersey: Lawrence Erlbaum Associates.

Zinn, C., Moore, J. & Core, M. (2005). Intelligent information presentation for tutoring systems. En O. Stock & M. Zancanaro (Eds.), *Multimodal intelligent information presentation. Serie: Text, Speech and Language Technology* (pp. 227-254). Kluver Academic Publishers.