Instituto Tecnológico de Costa Rica

Escuela de Ingeniería en Computadores

Paradigmas de Programación (CE1106)

Tarea #3:

TransLog

Profesor:

Marco Rivera Meneses

Estudiantes:

Javier Mora Masis – 2022101555

Allan Zheng Tang – 2021141102

Steven Aguilar Alvarez – 2024202865

II Semestre 2025

23/10/25

Contenido

1. Reglas Implementadas

TransLog implementa un conjunto de reglas lógicas organizadas en cuatro módulos principales:

- **database.pl:** Base de datos de palabras y traducciones
- **logic.pl:** Gramáticas DCG y reglas de parseo
- **bnf.pl:** Traducción de oraciones completas
- **main.pl:** Interfaz de usuario

Las reglas se dividen en dos categorías: reglas de reconocimiento (parsing) y reglas de traducción.

Reglas de Reconocimiento (Parsing)

1. Sintagma Nominal (SN)

Parsea un sintagma nominal en español e inglés con la estructura [Artículo] + [Adjetivos] + Sustantivo + [Adjetivos]. La regla reconoce secuencialmente cada componente, validando que el artículo concuerde en género y número con el sustantivo, y que todos los adjetivos sean válidos para el idioma. Esta regla es fundamental para descomponer la estructura de las oraciones.

2. Sintagma Verbal (SV)

Reconoce un sintagma verbal simple consistente en un verbo conjugado en tiempo presente. La regla busca una palabra que exista en la base de datos como verbo válido para el idioma correspondiente, sin validar persona o número adicionales durante el parsing.

3. Artículos Opcionales

Intenta reconocer un artículo al inicio del sintagma nominal, validando que tenga el género y número correcto. Si encuentra un artículo válido, lo captura; si no, devuelve un símbolo especial 'vacio' indicando ausencia de artículo, permitiendo construcciones sin determinante.

4. Adjetivos

Reconoce cero o más adjetivos consecutivos en la oración, funcionando recursivamente hasta encontrar una palabra que no sea adjetivo. Esta regla permite procesar oraciones con múltiples modificadores del sustantivo, manteniendo el orden de aparición.

5. Sustantivos

Identifica un sustantivo que existe en la base de datos con un género y número específicos. Consulta directamente la tabla de sustantivos para validar que la palabra sea reconocida en el idioma actual y captura sus características gramaticales.

6. Verbos

Reconoce un verbo conjugado en presente que existe en la base de datos del idioma. Consulta la tabla de verbos para validar que la forma verbal es una conjugación válida, sin necesidad de validar persona o número en esta etapa del parsing.

7. Oración completa

Parsea una oración como la concatenación de un sintagma nominal seguido de un sintagma verbal. Esta regla de nivel superior combina el reconocimiento de ambos componentes para validar la estructura completa de la oración.

Reglas de Validación y Concordancia

1. Concordancia Género-Número
Verifica que el artículo concuerde exactamente en género y número con el sustantivo del sintagma nominal. Si no hay artículo, la validación sucede automáticamente; si existe, consulta ambas palabras en la base de datos y confirma que sus atributos gramaticales coincidan.

2. Validación de Sintagmas Completos
Valida que un sintagma nominal sea completamente correcto incluyendo la existencia del artículo (si es obligatorio), validez de todos los adjetivos, presencia del sustantivo obligatorio, y concordancia general entre componentes. Esta regla garantiza la integridad estructural del sintagma.

Reglas de Traducción

1. Traducción de Sustantivos
Mapea sustantivos de un idioma a otro usando una tabla bidireccional en la base de datos. Para cada sustantivo en español existe su equivalente en inglés y viceversa, permitiendo búsquedas en ambas direcciones.

2. Traducción de Adjetivos
Traduce adjetivos considerando que en español existen variantes de género (pequeño, pequeña) que se mapean al mismo adjetivo en inglés (small). La regla maneja estas variaciones para mantener consistencia en la traducción.

3. Traducción de Verbos
Traduce formas verbales conjugadas específicas (corre, comes) a sus equivalentes en el idioma destino (runs, eats). La traducción es bidireccional y mantiene la forma conjugada, no traduce a infinitivos.

4. Traducción de Artículos
Traduce artículos de español a inglés perdiendo información de género y número (el, la, los, las → the). Para la traducción inversa, utiliza contexto de género y número del sustantivo para elegir el artículo correcto en español.

5. Traducción de SN
Traduce un sintagma nominal completo descomponiendo sus partes, traduciendo cada una individualmente (artículo con contexto, adjetivos, sustantivo) y reconstruyendo el sintagma en el idioma destino manteniendo la estructura y concordancia.

6. Traducción de SV
Traduce un sintagma verbal extrayendo el verbo conjugado, buscando su equivalente en la tabla de traducción y reconstruyendo el sintagma verbal con la forma conjugada correcta.

7. Traducción de Oraciones Completas
Traduce una oración completa traduciendo independientemente su sintagma nominal y su sintagma verbal, luego combinándolos en la estructura de oración destino.

Reglas de Interfaz

1. Traducción de Español a Inglés
Interfaz de alto nivel que parsea una lista de palabras como una oración en español, la traduce al inglés usando las reglas de traducción, y devuelve el resultado como una nueva lista de palabras en inglés.

2. Traducción de Inglés a Español
Interfaz de alto nivel que parsea una lista de palabras como una oración en inglés, la traduce al español usando las reglas de traducción, y devuelve el resultado como una nueva lista de palabras en español.

2. Estructuras de Datos Desarrolladas

1. oracion(SN, SV)

La estructura `oracion(SN, SV)` es la representación abstracta fundamental de una oración completa en TransLog. Está compuesta por dos componentes principales: el Sintagma Nominal (SN) que representa al sujeto de la oración, y el Sintagma Verbal (SV) que representa el predicado o acción. Esta estructura es generada por el algoritmo de parsing DCG como resultado de convertir una lista de palabras en una representación sintáctica estructurada. Por ejemplo, la oración "el perro corre" se representa como `oracion(sn(el, [], perro, []), sv(corre))`, donde el primer parámetro es el sintagma nominal y el segundo es el sintagma verbal.

La importancia de esta estructura radica en que es el punto de conexión entre el mundo de las palabras individuales y el mundo de las estructuras lógicas. Una vez que la oración ha sido parseada en esta forma, todos los algoritmos posteriores trabajan con esta estructura abstracta en lugar de listas de palabras, permitiendo un procesamiento modular y separado del sujeto y el predicado. Cuando llega el momento de traducir, la oración se descompone en sus dos componentes, se traduce cada uno independientemente, y luego se reconstruye una nueva estructura `oracion(SN_traducido, SV_traducido)` en el idioma destino.

2. sn(Articulo, Adjetivos_previos, Sustantivo, Adjetivos_posteriores)

El Sintagma Nominal es una estructura que encapsula todos los elementos que componen la parte nominal de una oración: el artículo opcional, los adjetivos que preceden al sustantivo, el sustantivo mismo, y los adjetivos que lo siguen. Esta estructura permite que cada componente sea identificado y procesado de manera independiente, facilitando la traducción granular. Por ejemplo, el sintagma nominal "la casa bonita grande" se representa como `sn(la, [], casa, [bonita, grande])`, donde "la" es el artículo, la lista vacía antes del sustantivo indica ausencia de adjetivos previos, "casa" es el sustantivo, y `[bonita, grande]` son los adjetivos posteriores.

La utilidad fundamental de esta estructura es que permite aplicar diferentes algoritmos a cada componente. El artículo puede ser traducido con consideración del contexto (género y número), cada adjetivo puede ser traducido individualmente, y el sustantivo puede ser traducido consultando su entrada en la base de datos. Además, esta estructura hace explícita la concordancia que debe existir entre artículo y sustantivo, permitiendo que el algoritmo de validación de concordancia verifique

fácilmente que el género y número sean iguales. Sin esta estructura, sería mucho más difícil mantener la coherencia gramatical durante la traducción.

3. sv(Verbo)

La estructura `sv(Verbo)` representa el Sintagma Verbal en su forma más simple, conteniendo únicamente un verbo conjugado. A diferencia del sintagma nominal que puede tener múltiples componentes opcionales, el sintagma verbal en la versión básica de TransLog es una estructura muy simple que contiene solo la acción de la oración. Por ejemplo, la oración "el perro corre" tiene un sintagma verbal representado como `sv(corre)`, donde "corre" es la tercera persona singular del presente indicativo del verbo "correr". Esta simplicidad es intencional y refleja las limitaciones autoimpuestas del sistema de mantener solo tiempos verbales presentes.

La importancia de mantener el verbo en una estructura dedicada es que permite diferenciar claramente entre el sujeto (sintagma nominal) y la acción (sintagma verbal), facilitando un procesamiento separado de ambos. Cuando se traduce una oración, el verbo puede ser traducido consultando la tabla `traducir_verbo()` en la base de datos, que devuelve la forma conjugada equivalente en el idioma destino. Por ejemplo, "corre" se traduce a "runs" en inglés, manteniendo la conjugación de tercera persona singular. Esta estructura, aunque simple, es crucial porque proporciona un punto de acceso claro y uniforme para la traducción del predicado de la oración.

4. sustantivo(idioma, palabra, genero, numero) / articulo(idioma, palabra, genero, numero)

Estas dos relaciones en la base de datos son estructuras fundamentales que almacenan información gramatical sobre sustantivos y artículos. Cada entrada tiene cuatro campos: el idioma (español o inglés), la palabra específica, el género gramatical (masculino o femenino), y el número (singular o plural). Por ejemplo, la entrada `sustantivo(esp, casa, fem, sing)` indica que la palabra "casa" en español es un sustantivo femenino singular, mientras que `articulo(esp, la, fem, sing)` indica que "la" es un artículo femenino singular. Estas relaciones se consultan miles de veces durante el procesamiento para obtener información crucial sobre las palabras.

La relevancia de estas estructuras radica en que son la base del sistema de concordancia gramatical. Sin estos atributos almacenados, sería imposible validar que un artículo concuerda correctamente con su sustantivo, o traducir adecuadamente un artículo en el idioma destino considerando el género y número del sustantivo. Cuando se traduce una oración como "la casa grande" del español al inglés, el algoritmo consulta estas relaciones para descubrir que "casa" es femenino singular, luego usa esa información para traducir "la" al artículo inglés correcto "the" (que no tiene variación de género). Sin estas estructuras de datos que asocian palabras con sus propiedades gramaticales, la concordancia sería imposible de mantener.

5. traducir_sustantivo/adjetivo/verbo/articulo(origen, destino) [bidireccional]

Estas relaciones de traducción son tablas de mapeo que conectan palabras entre idiomas, formando el corazón del mecanismo de traducción léxica de TransLog. Cada relación es bidireccional, conteniendo entradas en ambas direcciones: `traducir_sustantivo(perro, dog)` traduce español a inglés, mientras que `traducir_sustantivo(dog, perro)` traduce inglés a español.

Existen relaciones separadas para sustantivos, adjetivos, verbos y artículos, permitiendo que cada categoría gramatical tenga su propia tabla de traducción especializada. Por ejemplo, `traducir_verbo(corre, runs)` mapea la forma conjugada española a la forma conjugada inglesa, manteniendo la información de conjugación en ambas direcciones.

La importancia crucial de estas estructuras es que implementan la funcionalidad central de TransLog: la traducción de palabras individuales. Sin estas relaciones, no habría manera de convertir palabras de un idioma a otro. Cada componente de una oración (sustantivo, adjetivo, verbo, artículo) se traduce consultando la relación apropiada, obteniendo la palabra equivalente en el idioma destino. La bidireccionalidad garantiza que el sistema funcione tanto para traducir de español a inglés como en la dirección inversa, sin necesidad de código duplicado o lógica especial. Estas tablas contienen aproximadamente 1000 entradas que cubren el vocabulario completo soportado por TransLog, permitiendo que cualquier combinación válida de palabras pueda ser traducida correctamente.

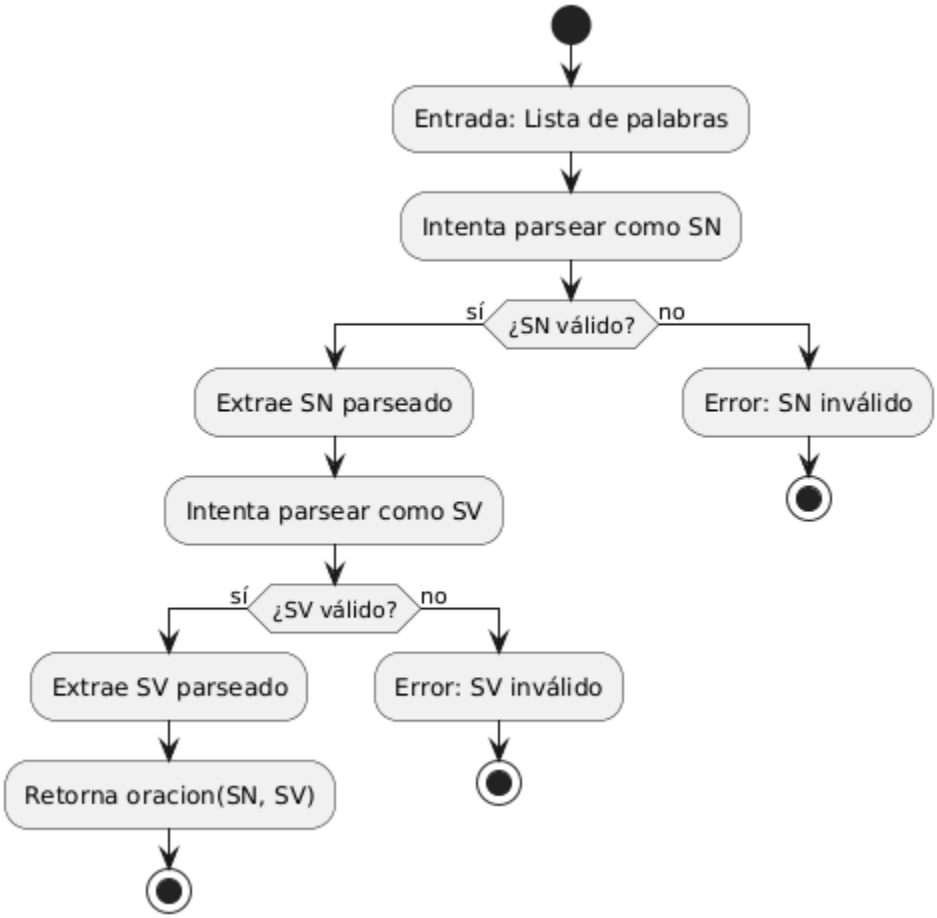
3. Algoritmos Desarrollados

1. Algoritmo de Parsing DCG (Definite Clause Grammar)

El algoritmo de parsing DCG es el responsable de convertir una lista de palabras en una estructura sintáctica reconocible. Recibe como entrada una secuencia de palabras y comienza por intentar parsear un sintagma nominal (SN), que constituye el sujeto de la oración. Utiliza las reglas gramaticales definidas mediante cláusulas DCG para descomponer la lista de palabras, extrayendo artículos opcionales, adjetivos, sustantivos y más adjetivos en el orden correcto según la gramática del idioma.

Una vez que el sintagma nominal ha sido parseado exitosamente, el algoritmo continúa intentando parsear un sintagma verbal (SV), que contiene la acción o predicado de la oración. Si ambos componentes se parsearon correctamente, el algoritmo retorna una estructura de árbol sintáctico representada como `oracion(SN, SV)`. Este árbol contiene toda la información necesaria sobre la estructura de la oración original, permitiendo que algoritmos posteriores trabajen con una representación abstracta en lugar de palabras individuales.

Si en cualquier momento el parsing falla, ya sea porque el sintagma nominal no es válido o el sintagma verbal no es reconocible, el algoritmo detiene el proceso y retorna un error. Esto es fundamental para mantener la integridad del sistema, asegurando que solo se traducen oraciones que se ajustan a la gramática predefinida. El mecanismo de backtracking de Prolog permite que el algoritmo intente diferentes caminos de parsing hasta encontrar el correcto o agotarlos todos.



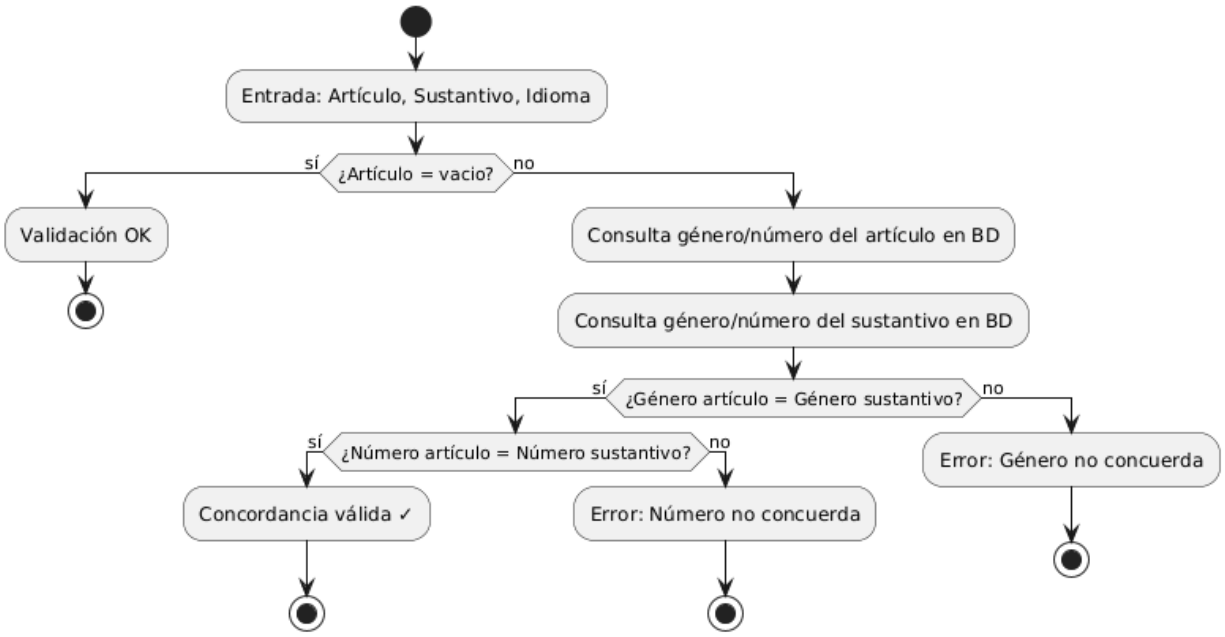
2. Algoritmo de Validación de Concordancia

El algoritmo de validación de concordancia garantiza que los elementos de una oración mantengan coherencia gramatical, específicamente verificando que el artículo concuerde en género y número con el sustantivo que acompaña. Cuando el artículo es vacío o no existe, el algoritmo inmediatamente considera que la validación es exitosa, ya que no hay concordancia que verificar. Sin embargo, cuando un artículo está presente, el algoritmo consulta la base de datos para obtener los atributos gramaticales del artículo y del sustantivo.

El proceso de verificación compara primero el género del artículo con el género del sustantivo. Si los géneros coinciden, el algoritmo procede a verificar que el número también sea igual. Por ejemplo, si el artículo es "la" (femenino singular) y el sustantivo es "casa" (femenino singular), ambos atributos coinciden y la concordancia es válida. Si alguno de estos atributos no coincide, como cuando se intenta usar "el" (masculino) con "casa" (femenino), el algoritmo detiene el proceso y retorna un error específico indicando si es el género o el número el que no concuerda.

Esta validación es crucial para mantener la coherencia gramatical de las oraciones traducidas, especialmente en la reconstrucción de sintagmas nominales en el idioma destino. Sin este algoritmo, el sistema podría generar oraciones gramaticalmente incorrectas como "la perro" o "el casa", que son aceptables sintácticamente pero incorrectas gramaticalmente. Al ejecutarse antes de la

reconstrucción final, este algoritmo asegura que todas las oraciones producidas por TransLog cumplan con las reglas de concordancia del idioma destino.

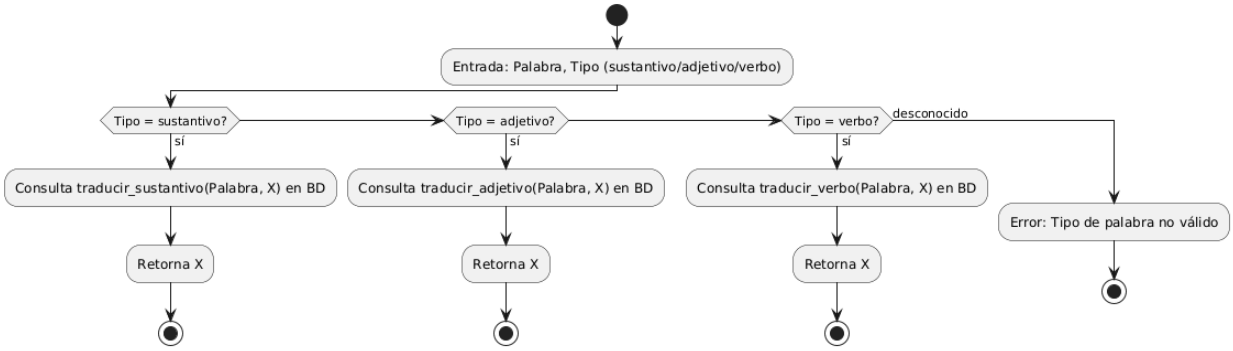


3. Algoritmo de Traducción de Componentes

El algoritmo de traducción de componentes es responsable de convertir palabras individuales de un idioma a otro consultando las tablas de traducción almacenadas en la base de datos. El algoritmo recibe como entrada una palabra específica y el tipo de palabra que es (sustantivo, adjetivo o verbo), y utiliza este tipo para determinar cuál tabla de traducción debe consultar. Esta estratificación por tipo de palabra es esencial porque permite mantener tablas de traducción separadas y eficientes para cada categoría gramatical.

Dependiendo del tipo de palabra, el algoritmo ejecuta diferentes consultas a la base de datos. Para sustantivos, consulta la relación `traducir_sustantivo(Palabra_origen, Palabra_destino)`, para adjetivos consulta `traducir_adjetivo()`, y para verbos consulta `traducir_verbo()`. Cada una de estas consultas busca una correspondencia exacta en la base de datos y retorna la traducción equivalente. Por ejemplo, si se consulta por el sustantivo "gato" en español, la función retorna "cat" en inglés; si se consulta por el adjetivo "grande", retorna "big".

Si la palabra no se encuentra en la tabla de traducción correspondiente, el algoritmo retorna un error, lo que indica que la palabra está fuera del vocabulario del sistema. Esto es un mecanismo de seguridad que previene que se traduzcan palabras desconocidas de manera incorrecta. El algoritmo también es bidireccional, funcionando tanto para traducir de español a inglés como en la dirección inversa, porque la base de datos contiene entradas en ambas direcciones para garantizar la bidireccionalidad completa del sistema.

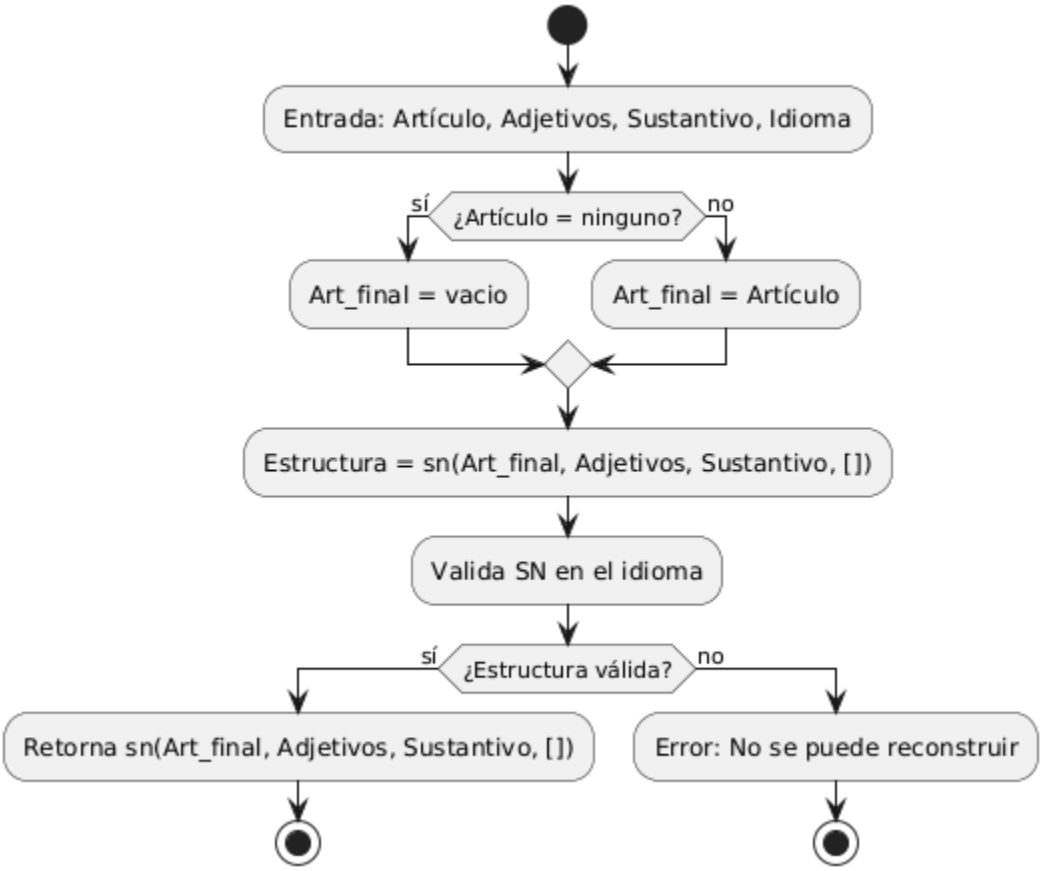


4. Algoritmo de Reconstrucción de Sintagmas

El algoritmo de reconstrucción de sintagmas toma componentes individuales que han sido traducidos (artículo traducido, adjetivos traducidos, sustantivo traducido) y los reúne en una estructura sintáctica válida del idioma destino. El algoritmo comienza verificando si el artículo traducido es "ninguno" (es decir, si no había artículo en la oración original). Si es así, establece el artículo final como "vacío"; de lo contrario, utiliza el artículo traducido tal como fue retornado por el algoritmo anterior.

Una vez que ha determinado el estado del artículo, el algoritmo construye una estructura de sintagma nominal con la forma `sn(Artículo_final, Adjetivos_traducidos, Sustantivo_traducido, [])`. Esta estructura es una representación abstracta del sintagma nominal que será válida en el idioma destino. Luego, el algoritmo valida que esta estructura sea correcta en el contexto del idioma destino, verificando que todos los componentes existan en la base de datos del idioma destino y que mantengan la concordancia gramatical apropiada. Si la estructura es válida, el algoritmo retorna el sintagma nominal reconstruido.

Si la estructura no puede ser validada, el algoritmo retorna un error, indicando que no es posible reconstruir un sintagma nominal válido con los componentes dados. Este mecanismo de validación es fundamental porque garantiza que el sintagma nominal reconstruido no solo tiene componentes traducidos correctamente, sino que también mantiene la coherencia gramatical requerida por el idioma destino. Por ejemplo, si se intenta reconstruir un sintagma nominal con una concordancia incorrecta entre artículo y sustantivo, este algoritmo detecta el error antes de que se produzca una oración incorrecta.



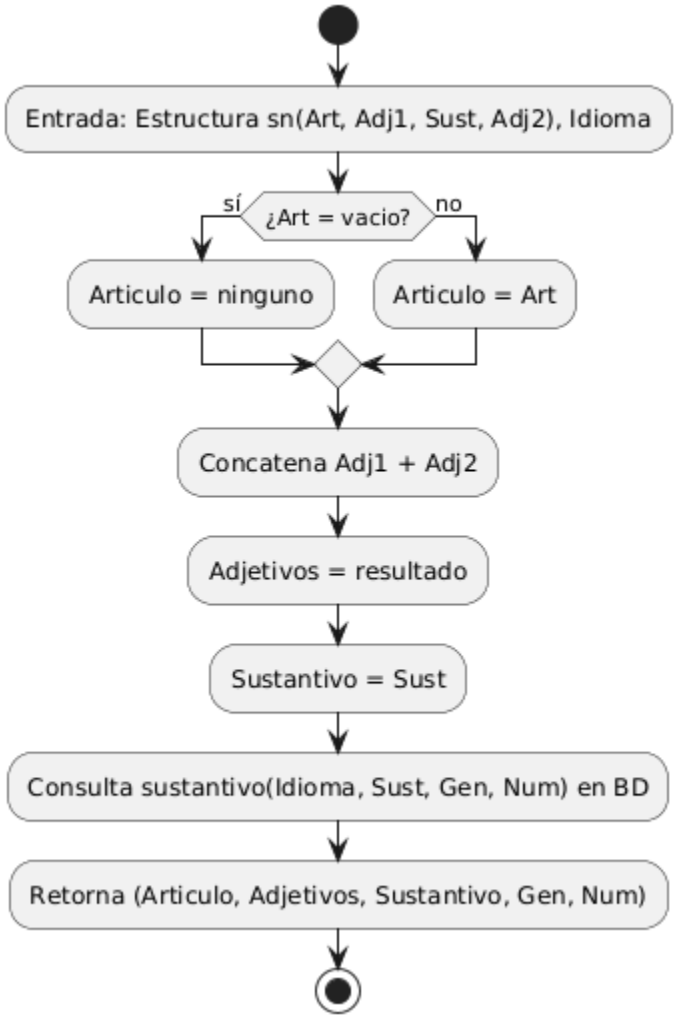
5. Algoritmo de Descomposición de Estructuras

El algoritmo de descomposición de estructuras realiza la operación inversa a la reconstrucción, extrayendo los componentes individuales de una estructura sintáctica para facilitar su procesamiento independiente. El algoritmo recibe como entrada una estructura de sintagma nominal en la forma `sn(Artículo, Adjetivos_previos, Sustantivo, Adjetivos_posteriores)` y el idioma asociado a esa estructura. Su objetivo es extraer y organizar cada componente de manera que pueda ser procesado por algoritmos posteriores.

El algoritmo comienza por inspeccionar el artículo: si es "vacío", lo convierte a "ninguno" para indicar que no hay artículo; si tiene un valor, lo mantiene tal como es. Luego concatena los adjetivos previos y posteriores en una lista única de adjetivos, y extrae el sustantivo. Finalmente, consulta la base de datos para obtener los atributos gramaticales del sustantivo, específicamente su género y número. Este paso es crucial porque estos atributos serán necesarios cuando se traduzca el artículo al idioma destino, ya que el artículo debe concordar con estos mismos atributos.

El algoritmo retorna una tupla con todos los componentes descompuestos: `(Artículo, Adjetivos, Sustantivo, Género, Número)`. Esta información estructurada es exactamente lo que necesitan los algoritmos de traducción posteriores para procesar cada componente de manera adecuada. Al separar la estructura en sus componentes fundamentales, el algoritmo permite que cada

parte sea traducida independientemente antes de ser recombinada en la estructura del idioma destino, facilitando un proceso de traducción modular y eficiente.



4. Problemas sin Solución

Problema 1: Verbos Auxiliares (do, does)

En inglés, las oraciones negativas e interrogativas requieren verbos auxiliares como "do" y "does": "the cat does not eat" en lugar de "the cat not eats", o "does the cat eat?" en lugar de "the cat eat?". El sistema actual traduce estas construcciones de manera simplificada pero gramaticalmente incorrecta en inglés. La complejidad reside en que los auxiliares se conjugan según la persona: "I do eat", "he/she/it does eat", mientras que el sistema solo mantiene género y número, no persona gramatical. Implementar esto requeriría redesñar completamente la estructura de datos de verbos, refactorizar la base de datos, y reescribir algoritmos de traducción verbal. Dada la escala de cambios necesarios y el objetivo educativo del proyecto, se priorizó mantener la simplicidad del sistema sobre la corrección gramatical en inglés.

Problema 2: Oraciones Complejas y Subordinadas

El sistema solo soporta oraciones simples con estructura fija `SN + SV`, rechazando oraciones complejas como "el perro que ladra corre" (subordinada relativa), "el perro corre y el gato salta" (coordinación), o "si llueve, me quedo en casa" (subordinada condicional). Las subordinadas requieren mantener contexto entre cláusulas, las coordinaciones requieren concordancia entre oraciones, y las condicionales requieren lógica temporal que TransLog no soporta. Implementar esto requeriría rediseñar completamente la gramática DCG para soportar recursión de oraciones, expandir la base de datos con nuevas categorías lingüísticas, e implementar análisis de referencias anafóricas entre cláusulas. Representaría un proyecto completamente nuevo, varias veces más complejo que TransLog actual, sacrificando significativamente la claridad pedagógica que es el objetivo educativo del sistema.

Problema 3: Estructura Fija de Oración (SN + SV)

El sistema requiere que todas las oraciones sigan exactamente la estructura `SN + SV` (Sintagma Nominal seguido de Sintagma Verbal). Aunque todas las palabras individuales estén en la base de datos, el sistema rechaza oraciones que no cumplan esta estructura: "el perro corre" funciona, pero "perro corre" (sin artículo), "corre perro" (orden invertido), "el perro" (sin SV), o "corre" (solo verbo) fallan. La gramática DCG está diseñada específicamente para reconocer `oracion(SN, SV)` y rechaza cualquier variación. Hacer flexible esta estructura requeriría redesñar completamente las reglas gramaticales para permitir artículos opcionales verdaderos, órdenes de palabras alternativas, y omisión de componentes, lo que abriría ambigüedad masiva en parsing. TransLog optó por mantener una gramática rígida para ser pedagógicamente clara, siendo este un trade-off aceptable entre flexibilidad y claridad educativa.

5. Plan de Actividades

Actividad	Descripción	Tiempo Estimado	Fecha de entrega	Responsable
SEMANA 1				
Investigación y Diseño de Arquitectura	Estudiar DCG en Prolog, definir alcance del traductor (qué estructuras soportará) y diseñar arquitectura general del sistema (módulos BD-Logic-BNF)	6 horas	11/10/25	TODOS
Implementación de Base de Datos (BD.pl)	Crear diccionario bilingüe con palabras clasificadas por categoría gramatical (sustantivos, verbos, adjetivos, determinantes, pronombres). Incluir funciones de consulta y búsqueda	8 horas	13/10/25	Steven y Allan
Diseño y Especificación de Gramática CFG	Diseñar reglas de gramática libre de contexto en notación BNF para ambos idiomas. Validar diseño en equipo antes de implementar	6 horas	14/10/25	Allan
Implementación de DCG para Parsing (Logic.pl - Parte 1)	Implementar Definite Clause Grammars para reconocer y descomponer sintagmas nominales (SN) en ambos idiomas	8 horas	15/10/25	Steven
Implementación de DCG para Verbos (Logic.pl - Parte 2)	Completar DCG para sintagmas verbales (SV) y oraciones completas. Integrar con SN	8 horas	16/10/25	Steven
SEMANA 2				
Checkpoint e Integración de Parsing	Probar exhaustivamente que las DCG reconocen estructuras correctamente. Ajustar diccionario si faltan palabras. Punto de sincronización del equipo	6 horas	17/10/25	TODOS
Algoritmo de Traducción (Logic.pl - Parte 3)	Implementar lógica que toma estructuras parseadas, traduce palabra por palabra desde el diccionario y reconstruye oración en idioma objetivo manteniendo orden gramatical correcto	10 horas	18/10/25	Allan
Interfaz de Usuario (BNF.pl)	Crear interfaz principal que: (1) detecta idioma de entrada, (2) invoca parsing, (3) ejecuta traducción, (4) formatea y muestra resultado	8 horas	19/10/25	Javier
Pruebas de Integración y Casos Edge	Ejecutar batería de pruebas completas. Manejar errores (palabras no encontradas, oraciones mal formadas). Pulir experiencia de usuario	6 horas	20/10/25	Javier
Documentación Técnica y Manual de Usuario	Completar toda la documentación requerida: reglas implementadas, estructuras de datos, algoritmos detallados, bitácora, problemas encontrados, manual de usuario, conclusiones y bibliografía	12 horas	21/10/25	TODOS

6. Problemas Encontrados

Problema 1: Desajuste entre Estructura de Gramática y Base de Datos

Un problema era que la gramática no usaba la misma estructura que la base de datos. En la base de datos las palabras tenían cuatro características: idioma, palabra, género y número. Pero la gramática intentaba usarlas con menos información, como si tuvieran solo idioma y palabra. Por eso Prolog no encontraba coincidencias reales y devolvía traducciones equivocadas. La solución fue hacer que la gramática use la misma forma que la base de datos, con los cuatro datos completos. Además, ahora la gramática usa las reglas de traducción de la base de datos para obtener la palabra en el otro idioma. Así, ambas partes del programa quedaron conectadas y las traducciones empezaron a funcionar correctamente.

7. Recomendaciones

Se recomienda agregar soporte para oraciones negativas mediante la incorporación de la palabra "no" como operador lógico que modifique la estructura del sintagma verbal. Implementar preguntas simples utilizando palabras interrogativas (cómo, cuándo, dónde) requeriría modificar la gramática para aceptar estas palabras al inicio de la oración. Agregar soporte para conjunciones (y, o, pero) permitiría traducir oraciones compuestas de manera limitada, combinando múltiples oraciones simples. Extender la conjugación verbal a otros tiempos (pasado, futuro, condicional) aumentaría significativamente la utilidad del sistema, aunque requeriría mayor complejidad en las reglas de traducción.

Se sugiere expandir considerablemente la base de datos de palabras, pasando de 1000 palabras aproximadamente a al menos 5000 para mejorar la cobertura del vocabulario. Implementar un módulo que permita agregar palabras dinámicamente durante la ejecución del sistema facilitaría su uso educativo y extensibilidad. Organizar la base de datos por categorías semánticas (animales, alimentos, acciones, etc.) permitiría mejor mantenimiento y análisis del contenido. Incluir información adicional como sinónimos y campos semánticos ayudaría a mejorar futuras versiones más sofisticadas del traductor.

Se recomienda crear una interfaz gráfica más intuitiva que reemplace la interfaz de línea de comandos actual, permitiendo visualizar el árbol sintáctico de la oración parseada para propósitos educativos. Implementar un sistema de corrección ortográfica que sugiera palabras similares cuando una palabra no sea encontrada en la base de datos mejoraría la experiencia del usuario. Agregar estadísticas de uso y un historial de traducciones permitiría analizar patrones de errores. Finalmente, crear una versión web del sistema haría que TransLog sea más accesible para estudiantes e investigadores interesados en programación lógica y procesamiento de lenguaje natural.

8. Conclusiones

TransLog ha demostrado ser un sistema viable para la traducción automática de oraciones simples entre español e inglés utilizando programación lógica pura. El sistema implementa exitosamente gramáticas libres de contexto (DCG) para el análisis sintáctico, permitiendo la descomposición

precisa de oraciones en sintagmas nominales y verbales. La arquitectura modular del proyecto facilita el mantenimiento y la extensión, con una clara separación entre la base de datos (database.pl), las reglas de lógica (logic.pl), la traducción (BNF.pl) y la interfaz de usuario (main.pl). El manejo correcto de concordancia de género y número demuestra la capacidad del sistema para aplicar reglas gramaticales complejas de manera consistente.

El sistema está limitado a oraciones simples en tiempo presente con estructura fija (Sintagma Nominal + Sintagma Verbal), lo que restringe significativamente su aplicabilidad a casos reales de traducción. La ausencia de soporte para negaciones, preguntas, subordinadas y oraciones complejas hace que TransLog sea principalmente un sistema educativo antes que un traductor práctico. Aunque la base de datos contiene aproximadamente 1000 palabras, cualquier palabra no registrada causará fallo en la traducción, limitando la cobertura del vocabulario. El sistema tampoco maneja tiempos verbales más allá del presente, contexto lingüístico, ni ambigüedades semánticas.

TransLog representa una contribución significativa para la comprensión de los paradigmas de programación lógica y declarativa. El proyecto demuestra cómo Prolog puede ser utilizado para resolver problemas de procesamiento de lenguaje natural mediante declaración de reglas lógicas en lugar de instrucciones procedurales. El uso de DCG muestra una aplicación práctica de la recursión, backtracking y unificación, conceptos fundamentales en programación lógica. Finalmente, el sistema sirve como caso de estudio efectivo para estudiantes que buscan comprender cómo los sistemas automáticos pueden descomponer, analizar y traducir lenguaje natural.

9. Bibliografía

Monferrer, T. E., Toledo, F., & Pacheco, J. (2001). *El lenguaje de programación Prolog*. Universidad Politécnica de Valencia.

Universidad Nacional Autónoma de México, Facultad de Estudios Superiores Cuautitlán. (s.f.). *PROLOG*. https://virtual.cuautitlan.unam.mx/intar/?page_id=212

Wielemaker, J., Schrijvers, T., Triska, M., & Lager, T. (2012). *SWI-Prolog*. *Theory and Practice of Logic Programming, 12*(1–2), 67–96. https://doi.org/10.1017/S1471068411000494