

Sprawozdanie z projektu: Spanish SVO Constructor

Przedmiot: Przetwarzanie Języka Naturalnego

Temat: Konstruktor zdań w strukturze SVO dla języka
hiszpańskiego

Autorzy: Robert Jacak, Maciej Jamróży

Technologie: Python 3, CustomTkinter

Architektura: Model-View-Controller (MVC)

8 lutego 2026

Spis treści

1	Wstęp	2
1.1	Cel projektu	2
1.2	Wybór języka naturalnego	2
1.3	Technologie	2
2	Architektura systemu	3
2.1	Podział na moduły	3
2.2	Przepływ danych	3
2.3	Mechanizm “Base vs Derived Form”	3
3	Algorytmy	4
3.1	Algorytm uzgadniania fraz nominalnych (Noun Phrase Agreement)	4
3.1.1	Faza 1: Infleksja rodzajowa (Gender Inflection)	4
3.1.2	Faza 2: Infleksja liczbowa (Number Inflection)	4
3.2	Algorytm koniugacji czasownika	5
3.2.1	Krok 1: Identyfikacja grupy koniugacyjnej	5
3.2.2	Krok 2: Rozgałęzienie według trybu	5
3.2.3	Krok 3: Priorytet form nieregularnych	6
3.2.4	Krok 4: Modyfikacje trybu	6
3.3	Algorytm asemblacji zdania	6
4	Prezentacja działania (Use Case)	7
4.1	Wzorzec Wizard (Kreator)	7
4.2	Przykładowy scenariusz	7
4.3	Panel diagnostyczny	8
5	Podsumowanie	8
5.1	Wnioski	8
5.2	Osiągnięte wymagania	9

1 Wstęp

1.1 Cel projektu

Celem projektu było zaprojektowanie i zaimplementowanie interaktywnego narzędzia desktopowego, które wizualizuje procesy morfologiczne zachodzące podczas konstruowania zdań w języku hiszpańskim w porządku SVO (Subject-Verb-Object, pol. Podmiot-Orzeczenie-Dopełnienie). Aplikacja realizuje paradygmat **Rule-Based NLP** (przetwarzanie języka naturalnego oparte na regułach), w którym każda transformacja morfologiczna jest jawnie zdefiniowana i możliwa do prześledzenia przez użytkownika.

1.2 Wybór języka naturalnego

Jako język docelowy wybrano **język hiszpański**. Decyzja ta była podyktowana bogactwem fleksyjnym tego języka w porównaniu z językiem angielskim. Hiszpański oferuje:

- **System rodzajów gramatycznych** (masculino / femenino) – każdy rzeczownik posiada przypisany rodzaj, który wymusza uzgodnienie formy przymiotnika i rodzaju.
- **Rozbudowaną koniugację czasowników** – trzy grupy koniugacyjne (-ar, -er, -ir) z odrębnymi zestawami końcówek dla sześciu osób gramatycznych, trzech czasów, pięciu trybów oraz licznymi formami nieregularnymi.
- **Uzgodnienie wewnątrz frazy nominalnej** (Noun Phrase Agreement) – rodzajnik, rzeczownik i przymiotnik muszą ze sobą współgrać pod względem rodzaju i liczby.

Te cechy sprawiają, że hiszpański stanowi znacznie lepszy materiał dydaktyczny do demonstracji algorytmów morfologicznych niż język angielski, który jest w dużej mierze językiem analitycznym o minimalnej fleksji.

1.3 Technologie

Projekt został zrealizowany w języku **Python 3** z wykorzystaniem biblioteki **CustomTkinter** jako frameworka GUI. CustomTkinter zapewnia nowoczesny, ciemny motyw wizualny (dark mode) przy zachowaniu prostoty programowania charakterystycznej dla Tkinter. Architektura aplikacji jest oparta na wzorcu **MVC (Model-View-Controller)**, co zapewnia separację warstwy danych, logiki biznesowej i prezentacji.

2 Architektura systemu

2.1 Podział na moduły

System składa się z trzech modułów źródłowych, z których każdy pełni ściśle określoną rolę architektoniczną:

Moduł	Rola w MVC	Odpowiedzialność
<code>data.py</code>	Model (dane)	Baza wiedzy lingwistycznej – słowniki rzeczowników, przymiotników, czasowników, rodzajników oraz tablice końcówek koniugacyjnych.
<code>grammar.py</code>	Model (logika)	Silnik regułowy – algorytmy uzgadniania fraz nominalnych, koniugacji czasowników i asemblacji zdań.
<code>main.py</code>	View + Controller	Warstwa prezentacji (GUI w CustomTkinter) oraz kontroler obsługujący zdarzenia użytkownika i delegujący przetwarzanie do warstwy modelu.

Tabela 1: Struktura modułowa projektu

2.2 Przepływ danych

Przepływ sterowania przebiega następująco:

1. Użytkownik dokonuje selekcji parametrów w interfejsie graficznym (kontroler w `main.py`).
2. Po kliknięciu przycisku "Build Sentence" kontroler ekstrahuje wartości z widżetów i przekazuje je do metody orkiestrującej `SentenceBuilder.build_full()` w module `grammar.py`.
3. Silnik regułowy pobiera dane leksykalne z `data.py`, aplikuje algorytmy morfologiczne i zwraca parę: (wygenerowane zdanie, pełny log transformacji).
4. Kontroler wyświetla wynikowe zdanie oraz szczegółowy log w panelu rezultatu.

2.3 Mechanizm “Base vs Derived Form”

Kluczową decyzją projektową jest przechowywanie wszystkich jednostek leksykalnych w **formie bazowej (citation form)**:

- **Rzeczowniki** są przechowywane z kluczem odpowiadającym formie podstawowej (np. "gato"), a słownik zawiera gotowe formy liczby pojedynczej i mnogiej ("gato" / "gatos"), ponieważ pluralizacja w hiszpańskim jest nieregularna (np. "pez" → "peces", "país" → "países").

- **Przymiotniki** przechowywane są wyłącznie w formie bazowej masculino singular (np. "rojo"), a formy pochodne (np. "roja", "rojos", "rojas") generowane są algorytmicznie w czasie rzeczywistym przez silnik uzgadniania.
- **Czasowniki** przechowywane są w formie bezokolicznika (np. "hablar"), a formy odmienione wytwarzane są przez silnik koniugacji na podstawie grupy koniugacyjnej, osoby, czasu i trybu.

Takie podejście minimalizuje redundancję danych – zamiast przechowywania setek prekalkulowanych form, system przechowuje jedynie formy bazowe i reguły transformacji, co jest zgodne z paradygmatem **generatywnej morfologii regułowej**.

3 Algorytmy

3.1 Algorytm uzgadniania fraz nominalnych (Noun Phrase Agreement)

Algorytm uzgadniania, zaimplementowany w klasie **AgreementEngine**, odpowiada za infleksję przymiotników tak, aby ich forma gramatyczna była zgodna z rodzajem i liczbą rzeczownika, do którego się odnoszą. Algorytm działa w dwóch fazach sekwencyjnych.

3.1.1 Faza 1: Infleksja rodzajowa (Gender Inflection)

System klasyfikuje przymiotniki w trzy klasy morfologiczne i stosuje odpowiednią regułę:

Klasa	Warunek rozpoznania	Reguła	Przykład
Klasa -o	Forma bazowa kończy się na -o	M: bez zmian; F: zamiana -o na -a	rojo → roja
Klasa -e	Forma bazowa kończy się na -e	Forma inwariantna – brak zmiany dla rodzaju	grande → grande
Klasa spółgłoskowa	Forma bazowa kończy się spółgłoską	Forma inwariantna – brak zmiany dla rodzaju	feliz → feliz

Dodatkowo system obsługuje **formy nieregularne** (np. "bueno" / "buena"), które są przechowywane jako jawne mapowania w polu `irregular_forms` i mają priorytet nad regułami algorytmicznymi.

3.1.2 Faza 2: Infleksja liczbowa (Number Inflection)

Po ustaleniu formy rodzajowej algorytm aplikuje reguły pluralizacji:

Warunek końcówki formy pojedynczej	Reguła	Przykład
Kończy się samogłoską (a, e, i, o, u)	Dodaj -s	roja → rojas
Kończy się na -z	Zamień -z na -ces	feliz → felices
Kończy się inną spółgłoską	Dodaj -es	azul → azules

Wynikiem działania algorytmu jest w pełni odmieniona forma przymiotnika oraz szczegółowy log dokumentujący każdy krok transformacji. Przykładowo, dla wejścia `base="rojo"`, `gender="F"`, `number="plural"` algorytm produkuje ścieżkę: `rojo` → `roja` (infleksja rodzajowa) → `rojas` (infleksja liczbowa).

3.2 Algorytm koniugacji czasownika

Algorytm koniugacji, zaimplementowany w klasie `ConjugationEngine`, przetwarza bezokolicznik czasownika na formę odmienioną na podstawie trzech parametrów: osoby gramatycznej, czasu i trybu. System obsługuje trzy czasy (Presente, Preterito, Futuro) oraz pięć trybów (Afirmativo, Negativo, Interrogativo, Imperativo, Condicional).

3.2.1 Krok 1: Identyfikacja grupy koniugacyjnej

System rozpoznaje grupę koniugacyjną na podstawie dwuliterowej końcówki bezokolicznika:

Końcówka	Grupa	Przykładowe czasowniki
-ar	I	hablar, caminar, comprar
-er	II	comer, beber, correr
-ir	III	vivir, escribir, abrir

Rdzeń (stem) czasownika uzyskiwany jest przez obcięcie dwóch ostatnich znaków bezokolicznika (np. "hablar" → "habl").

3.2.2 Krok 2: Rozgałęzienie według trybu

Algorytm realizuje odmienną ścieżkę przetwarzania w zależności od wybranego trybu:

Tryb Imperativo (Rozkazujący): Koniugacja imperatywna wymaga odrębnego zestawu osób gramatycznych (tu, usted, nosotros, vosotros, ustedes) i posiada własną tablicę końcówek. System najpierw sprawdza, czy dany czasownik posiada nieregularne formy imperatywne – jeśli tak, zwraca formę z tablicy nieregularnej; w przeciwnym razie łączy rdzeń z końcówką regularną.

Tryb Condicional (Przypuszczający): Tryb warunkowy stosuje **odrębną strategię tworzenia formy** – końcówki dodawane są nie do rdzenia, lecz do pełnego bezokolicznika. Jest to zgodne z rzeczywistą regułą gramatyczną języka hiszpańskiego, gdzie futuro i condicional budowane są na bazie infinitivo. Końcówki trybu warunkowego (-ía, -ías, -ía, -íamos, -íais, -ían) są wspólne dla wszystkich trzech grup koniugacyjnych. Przykładowo: "comer" + ía" = "comería" (jadłby/jadłaby).

Czasy standardowe (Presente, Preterito, Futuro) w trybach Afirmativo/Negativo/Interrogativo: Dla czasu Futuro stosowana jest ta sama strategia co dla Condicional – końcówki dodaje się do pełnego bezokolicznika. Dla czasów Presente i Preterito końcówki dodawane są do rdzenia, przy czym każda grupa koniugacyjna posiada odrębny zestaw końcówek dla sześciu osób gramatycznych.

3.2.3 Krok 3: Priorytet form nieregularnych

Na każdym etapie algorytm najpierw sprawdza słownik form nieregularnych dla danego czasownika. Jeżeli dla konkretnej kombinacji (czas, osoba) istnieje wpis w tablicy *irregular*, jest on zwracany bezpośrednio, z pominięciem reguł generatywnych. System przechowuje pełne tablice nieregularne dla 10 najczęściej używanych czasowników nieregularnych (ser, estar, ir, tener, hacer, poder, querer, decir, saber, venir).

3.2.4 Krok 4: Modyfikacje trybu

Po uzyskaniu formy odmienionej algorytm aplikuje transformacje zależne od trybu:

- **Negativo:** prepend "no " przed formą (np. "habla" → "no habla").
- **Interrogativo:** obsługa na etapie asemblacji zdania (znaki ¿...?).
- **Afirmativo:** forma bez modyfikacji.

3.3 Algorytm asemblacji zdania

Klasa *SentenceBuilder* orkiestruje cały proces budowania zdania SVO. Metoda *build_full()* sekwencyjnie:

1. Buduje frazę nominalną podmiotu (determiner + noun + adjective) z uzgodnieniem.
2. Koniuguje czasownik.
3. Buduje frazę nominalną dopełnienia (opcjonalną).
4. Składa elementy w porządku SVO i aplikuje reguły interpunkcji zależne od trybu:
 - Interrogativo: ¿...?

- Imperativo: ¡...!
- Pozostałe: zdanie oznajmujące z kropką.

Każdy etap generuje szczegółowy log, który jest prezentowany użytkownikowi w panelu wynikowym.

4 Prezentacja działania (Use Case)

4.1 Wzorzec Wizard (Kreator)

Interfejs graficzny został zaprojektowany zgodnie ze wzorcem **Wizard Pattern** – użytkownik prowadzony jest przez kolejne, logicznie uporządkowane etapy konstrukcji zdania. Interfejs jest podzielony na cztery numerowane sekcje, które odpowiadają naturalnej kolejności budowania zdania SVO:

Krok 1 – SUBJECT (Podmiot): Użytkownik wybiera rodzaj gramatyczny (M/F) za pomocą segmentowanego przycisku, co dynamicznie filtruje listę dostępnych rzeczowników. Następnie wybiera liczbę (singular/plural), konkretny rzeczownik z listy rozwijanej, opcjonalny przymiotnik oraz typ rodzajnika (Definido, Indefinido, Demostrativo, Posesivo lub Ninguno). Panel podglądu (Preview) na żywo wyświetla formę rodzajnika uzgodnioną z wybranym rodzajem i liczbą.

Krok 2 – VERB (Czasownik): Użytkownik wybiera czasownik z listy rozwijanej (wyświetlanej w formacie **bezokolicznik (znaczenie)**), osobę gramatyczną, tryb i czas. Interfejs reaguje dynamicznie na zmianę trybu – po wybraniu trybu Imperativo lista osób przełącza się na formy imperatywne (tu, usted, nosotros, vosotros, ustedes), a selektor czasu zostaje dezaktywowany. Analogicznie, po wybraniu trybu Condicional selektor czasu jest dezaktywowany, ponieważ tryb warunkowy nie wymaga wyboru czasu.

Krok 3 – OBJECT (Dopełnienie): Konfiguracja analogiczna do podmiotu, z dodatkową opcją (ninguno) pozwalającą na pominięcie dopełnienia (zdania nieprzechodnie).

Krok 4 – RESULT (Wynik): Po kliknięciu przycisku "Build Sentence / Zbuduj zdanie" aplikacja wyświetla wygenerowane zdanie oraz szczegółowy log algorytmiczny dokumentujący każdą transformację morfologiczną.

4.2 Przykładowy scenariusz

Poniżej przedstawiono scenariusz demonstracyjny ilustrujący pełen przepływ pracy:

1. **Podmiot:** Użytkownik wybiera rodzaj M, liczbę **singular**, rzeczownik **niño** (boy), przymiotnik **alto** (tall), rodzajnik **Definido**. System buduje frazę: "el niño alto".
2. **Czasownik:** Użytkownik wybiera czasownik **comer** (to eat), osobę 3s (el/ella/usted), tryb **Condicional**. Selektor czasu zostaje automatycznie dezaktywowany. System koniuguje: "comer" + "ía" = "comería".
3. **Dopełnienie:** Użytkownik wybiera rodzaj F, liczbę **plural**, rzeczownik **manzana** (apple), przymiotnik **rojo** (red), rodzajnik **Indefinido**. System buduje frazę z uzgodnieniem: rodzajnik "unas" (F/plural), rzeczownik "manzanas", przymiotnik "rojo" → "roja" (gender) → "rojas" (number). Wynik: "unas manzanas rojas".
4. **Wynik:** System asembluje zdanie: "El niño alto comería unas manzanas rojas." i wyświetla pełny log transformacji w panelu wynikowym.

4.3 Panel diagnostyczny

Panel wynikowy wyświetla wygenerowany log podzielony na cztery sekcje (SUBJECT, VERB, OBJECT, SENTENCE ASSEMBLY), gdzie każda transformacja jest opisana w formacie:

```
Adjective base: rojo (red)
Gender: rojo -> roja (-o class: masculine -o changed to feminine -a
)
Number: roja -> rojas (vowel ending: +s)
```

Taki format zapewnia pełną **wyjaśnialność** (explainability) – użytkownik może prześledzić dokładną ścieżkę algorytmiczną od formy bazowej do formy wynikowej.

5 Podsumowanie

5.1 Wnioski

Projekt "Spanish SVO Constructor" realizuje paradygmat **Rule-Based NLP**, zapewniając pełną wyjaśnialność reguł gramatycznych dla ucznia języka hiszpańskiego. W odróżnieniu od podejść statystycznych (np. modele neuronowe), system regułowy pozwala na:

- **Transparentność** – każda transformacja jest jawnie udokumentowana w logu diagnostycznym.
- **Przewidywalność** – te same parametry wejściowe zawsze produkują identyczny wynik.

- **Dydaktyczność** – użytkownik może obserwować, jak system stosuje reguły gramatyczne krok po kroku.

System obsługuje 34 rzeczowniki (17 rodzaju męskiego, 17 żeńskiego), 31 przymiotników (w trzech klasach morfologicznych plus formy nieregularne), 22 czasowniki (12 regularnych -ar, 7 regularnych -er, 5 regularnych -ir oraz 10 nieregularnych), 7 typów rodzajników oraz pełną kombinatorykę 6 osób gramatycznych, 3 czasów i 5 trybów – w tym wymagany tryb Condicional.

5.2 Osiągnięte wymagania

- Zaimplementowano algorytm uzgadniania fraz nominalnych z obsługą trzech klas przymiotników oraz form nieregularnych.
- Zaimplementowano algorytm koniugacji z pełną obsługą trybu Condicional (przypuszczającego), Imperativo (rozkazującego) oraz trzech trybów standardowych.
- Zastosowano wzorzec Wizard Pattern prowadzący użytkownika przez sekwencyjne etapy budowania zdania.
- Zapewniono panel diagnostyczny z pełnym logiem transformacji morfologicznych.
- Architektura MVC zapewnia separację danych (`data.py`), logiki (`grammar.py`) i prezentacji (`main.py`).