Proyecto de Curso

Daniel Rosas, Sebastian Forero, David Hernández, Jose David Ontiveros
Estructuras de datos
Pontificia Universidad Javeriana

***Abstract*—Este documento describe el desarrollo de un sistema de apoyo interactivo para el juego Scrabble, destinado a asistir a los jugadores en la identificación de palabras con base en las letras disponibles y las condiciones del tablero. Se presenta un enfoque innovador que incluye la inicialización y manipulación de un diccionario de palabras, búsqueda optimizada de palabras con prefijos y sufijos, y un algoritmo de combinaciones de letras. El sistema propuesto no solo busca mejorar la experiencia de juego, sino también servir como herramienta pedagógica en el aprendizaje del lenguaje.**

***Index Terms*—Scrabble Sistema de apoyo interactivo, Identificación de palabras, Diccionario de palabras, Búsqueda de palabras, Aprendizaje del lenguaje**

I. Introduction

Scrabble es un juego de palabras ampliamente reconocido y apreciado tanto por su naturaleza competitiva como educativa. En este juego, la habilidad para construir palabras y la estrategia en el uso del tablero son cruciales para el éxito. Sin embargo, jugadores de todos los niveles pueden enfrentarse al desafío de optimizar sus jugadas. Para abordar esto, hemos desarrollado un sistema de apoyo que utiliza algoritmos avanzados para sugerir las mejores posibles palabras a partir de las letras disponibles. Además, el sistema incluye funcionalidades para evaluar la puntuación de palabras potenciales y admite interacción a través de una consola de comandos, facilitando así una experiencia de usuario fluida y educativa. Este trabajo detalla el diseño, la implementación y las capacidades del sistema, y discute su potencial para transformar el juego de Scrabble en una herramienta educativa más poderosa.

II. Diseño del Sistema

El diseño del sistema de apoyo interactivo para Scrabble se fundamenta en una arquitectura modular que comprende un procedimiento principal y múltiples operaciones auxiliares. Estas operaciones se implementan mediante comandos específicos que el usuario puede invocar a través de una interfaz de consola.

A. Procedimiento Principal

El procedimiento principal actúa como el núcleo del sistema, gestionando la inicialización del diccionario, la recepción de las entradas del usuario y la coordinación de las operaciones auxiliares.

Entradas:

- Letras disponibles: El usuario introduce las siete letras disponibles en su atril.
- Estado del tablero: Se introducen las letras ya colocadas en el tablero, junto con su posición.

Salidas:

- Lista de palabras posibles: Basándose en las entradas, el sistema proporciona una lista de palabras válidas.
- Puntuaciones estimadas: Para cada palabra sugerida, se calcula y muestra una puntuación basada en la ubicación potencial en el tablero y los modificadores de puntuación aplicables.

Condiciones:

- Las letras proporcionadas deben conformar palabras válidas dentro del diccionario del juego.
- Las palabras sugeridas deben encajar en el espacio disponible en el tablero, respetando las palabras ya colocadas.

B. Operaciones Auxiliares

Las operaciones auxiliares son comandos que el usuario puede ejecutar para realizar tareas específicas, como la manipulación de diccionarios y la evaluación de palabras.

1) Comando: Inicializar Diccionario: **Entrada:** Ruta al archivo del diccionario. **Salida:** Confirmación de que el diccionario ha sido cargado. **Condiciones:** El archivo del diccionario debe estar presente y ser legible.

2) Comando: Búsqueda por Prefijo/Sufijo: **Entrada:** Prefijo/Sufijo especificado por el usuario. **Salida:** Lista de palabras que comienzan/terminan con el prefijo/sufijo dado. **Condiciones:** El prefijo/sufijo debe existir en el diccionario.

3) Comando: Calcular Puntuación: **Entrada:** Palabra a evaluar y su posición potencial en el tablero. **Salida:** Puntuación calculada de la palabra basada en su ubicación y letras. **Condiciones:** La palabra debe ser válida y la posición debe estar libre en el tablero.

Cada uno de estos comandos se integra dentro del procedimiento principal para proporcionar una experiencia de usuario fluida y efectiva, asegurando que el sistema no solo asista en la formación de palabras, sino también en el aprendizaje y la estrategia del juego de Scrabble.

III. TADs

Dentro del desarrollo del sistema de soporte para el juego Scrabble, se han definido varios Tipos Abstractos de Datos (TADs) para encapsular la lógica y los datos de las entidades principales del juego. A continuación, se detallan los TADs clave y su funcionalidad.

A. TAD Diccionario

El TAD *Diccionario* representa la colección de todas las palabras válidas que pueden usarse en el juego.

Atributos:

- *palabras*: Una lista de objetos *Palabra* que contienen las palabras.

Métodos:

- *Diccionario()*: Constructor que inicializa un diccionario vacío.
- *getPalabras()*: Devuelve la lista de palabras del diccionario.
- *agregarPalabra(Palabra palabra)*: Agrega una nueva palabra al diccionario.
- *verificarPalabra(string palabra)*: Verifica si una palabra es válida.

B. TAD Letra

El TAD *Letra* gestiona la información individual de cada letra utilizada en el juego, incluyendo su puntuación.

Atributos:

- *puntaje*: Puntuación asociada a la letra.
- *valor*: Carácter que representa la letra.

Métodos:

- *Letra(char v)*: Constructor que inicializa la letra con un valor y puntaje.
- *getPuntaje()*: Retorna el puntaje de la letra.
- *setPuntaje(int puntaje)*: Establece el puntaje de la letra.

C. TAD ProcesadorComandos

El TAD *ProcesadorComandos* proporciona la lógica para interpretar y ejecutar los comandos ingresados por el usuario.

Métodos:

- *inicializarDiccionario(string rutaDiccionario)*: Carga palabras desde un archivo.
- *puntajePalabra(string palabra)*: Calcula el puntaje de una palabra dada.

D. TAD Palabras

El TAD *Palabras* encapsula las operaciones relacionadas con la manipulación de palabras en el juego.

Atributos:

- *letras*: Lista de *Letra* que componen la palabra.
- *puntaje*: Puntuación total de la palabra.

Métodos:

- *Palabra(string cadena)*: Construye una palabra a partir de una cadena de caracteres.
- *getLetras()*: Retorna las letras que componen la palabra.

E. TAD Consola

El TAD *Consola* maneja la interacción con el usuario a través de la línea de comandos.

Métodos:

- *recibirComando(char* comando)*: Procesa el comando ingresado por el usuario.
- *ayuda()*: Muestra los comandos disponibles.

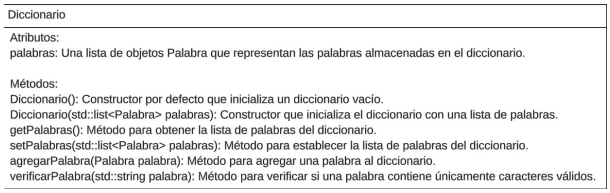

Fig. 1. TAD Diccionario

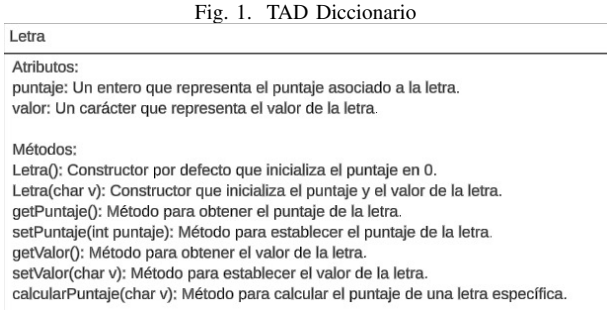

Fig. 2. TAD Letra

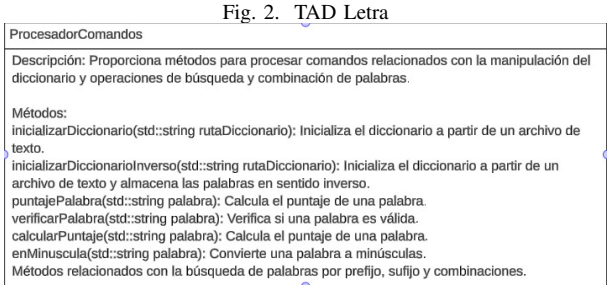

Fig. 3. TAD ProcesadorComandos

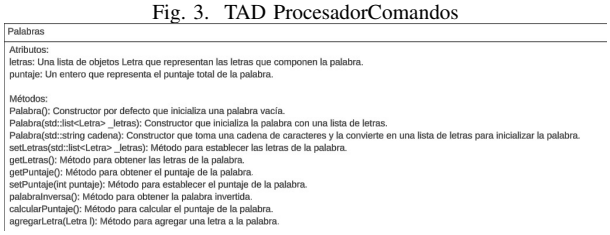

Fig. 4. TAD Palabra

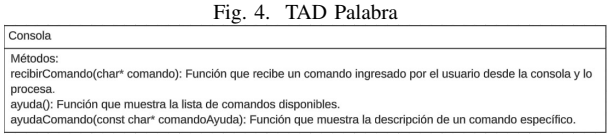

Fig. 5. TAD Consola

Caso de Prueba	Entrada	Resultado Esperado	Resultado Obtenido
Inicializar diccionario	Ruta valida al archivo	Diccionario cargado correctamente	Diccionario cargado correctamente
Agregar Palabra	Palabra valida "Pescado"	QUARK' añadida exitosamente	QUARK' añadida exitosamente
Verificar Palabra	Palabra invalida "Elefante"	Error: palabra inválida	Error: palabra inválida

IV. Plan de Pruebas

La siguiente tabla presenta el plan de pruebas diseñado para evaluar el sistema. Cada operación se somete a una serie de casos de prueba, comparando los resultados obtenidos con los esperados y proporcionando un análisis en caso de

discrepancias.

Caso de Prueba	Entrada	Resultado Esperado	Resultado Obtenido
Inicializar diccionario	Ruta valida al archivo	Diccionario cargado correctamente	Diccionario cargado correctamente
Agregar Palabra	Palabra valida "Pescado"	QUARK' añadida exitosamente	QUARK' añadida exitosamente
Verificar Palabra	Palabra invalida "Elefante"	Error: palabra inválida	Error: palabra inválida

V. Resultados Obtenidos

References

[1] J. Autor, *Título del libro*, 1ra ed. Ciudad: Editorial, Año.

[2] A. Autor y B. Coautor, "Título del artículo," *Nombre de la Revista*, vol. xx, no. xx, pp. xx–xx, mes, año.

[3] C. Autor, "Título del artículo de conferencia," en *Nombre de la Conferencia*, Ciudad, País, año, pp. xx–xx.