



INTRODUCCIÓN A SISTEMAS EXPERTOS

ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS

SI-881, INTELIGENCIA ARTIFICIAL

1. Edgard Reynaldo Chambe Torres, 0009-0009-7272-8651

2. Brant Antony Chata Choque, 0009-0004-6597-6837

Docente:

Dr. Oscar J. Jimenez Flores

Orcid:

0000-0002-7981-8467

Semestre:

UPT-EPIS, 2024-I

Resumen: Los sistemas expertos son programas computacionales diseñados para emular la sabiduría y juicio de un experto humano en un campo específico, lo que ha generado un impacto significativo en áreas como medicina, ingeniería y gestión empresarial al mejorar la eficiencia, reducir costos y preservar conocimientos. Estos sistemas, fundamentados en conceptos como la base de conocimientos y el motor de inferencia, son capaces de analizar grandes cantidades de datos y ofrecer soluciones precisas para problemas complejos. A pesar de su efectividad, presentan limitaciones en cuanto a su capacidad para aprender y adaptarse como otros sistemas de inteligencia artificial.

Palabras clave: Sistemas expertos, inteligencia artificial, aplicaciones en análisis.

1. Introducción

Los sistemas expertos representan un hito en la intersección entre la computación y la sabiduría humana. Estos programas informáticos, diseñados para emular el conocimiento y el juicio de un experto humano en un campo específico, han revolucionado diversas áreas.

Los sistemas expertos son programas que reproducen el proceso intelectual de un experto humano en un campo particular, pudiendo mejorar su productividad, ahorrar tiempo y dinero, conservar sus valiosos conocimientos y difundirlos más fácilmente. Antes de la aparición del ordenador, el hombre ya se preguntaba si se le arrebataría el privilegio de razonar y pensar. En la actualidad existe un campo dentro de la inteligencia artificial al que se le atribuye esa facultad: el de los sistemas expertos.

Estos sistemas permiten la creación de máquinas que razonan como el hombre, restringiéndose a un espacio de conocimientos limitado. En teoría pueden razonar siguiendo los pasos que seguiría un experto humano (médico, analista, empresario, etc.) para resolver un problema concreto. Este tipo de modelos de conocimiento por ordenador ofrece un extenso campo de posibilidades en resolución de problemas y en aprendizaje.

Su uso se extenderá ampliamente en el futuro, debido a su importante impacto sobre los negocios y la industria. Al combinar bases de conocimientos estructuradas con algoritmos de inferencia, los sistemas expertos son capaces de analizar datos, realizar deducciones y ofrecer soluciones precisas en campos tan variados como la medicina, la ingeniería, la gestión empresarial y más. Su capacidad para procesar enormes volúmenes de información y su habilidad para aprender y evolucionar con el tiempo los convierten en herramientas indispensables para abordar problemas complejos en el mundo contemporáneo.

2. Fundamento Teorico

Los sistemas expertos demuestran una eficacia notable al enfrentarse a la tarea de analizar grandes volúmenes de información, interpretarla y ofrecer recomendaciones basadas en dicho análisis.

Un ejemplo ilustrativo de su utilidad se encuentra en el ámbito del análisis financiero, donde estos sistemas pueden evaluar oportunidades de inversión, tomando en consideración los datos financieros de un cliente y sus objetivos específicos.

En el ámbito de la detección y reparación de fallos en equipos electrónicos, se recurre a los sistemas expertos de diagnóstico y depuración. Estos sistemas plantean una serie de preguntas para recabar los datos necesarios y así llegar a una conclusión sobre el problema.

Posteriormente, recomiendan las acciones apropiadas para solucionar los inconvenientes detectados. Este tipo de sistemas no solo se aplican en la electrónica, sino que también se utilizan en medicina, como los sistemas MYCIN y PUFF, así como en la localización de problemas en sistemas informáticos complejos y de gran envergadura.

Los sistemas expertos, dentro del ámbito de la inteligencia artificial, se enfocan en desarrollar programas informáticos capaces de imitar el conocimiento y el razonamiento de un experto humano en un área específica. Su estructura teórica se sustenta en varios conceptos clave:

- 1.Base de conocimientos: Es el componente central donde se almacena toda la información relevante sobre el dominio en estudio, incluyendo hechos, reglas y relaciones causales.
- 2.Motor de inferencia: Responsable de procesar la información de la base de conocimientos y aplicar reglas lógicas para deducir conclusiones.
- 3.Interfaz de usuario: Facilita la interacción entre el usuario y el sistema experto, permitiendo ingresar datos, hacer consultas y recibir resultados.
- 4.Adquisición de conocimientos: Proceso para capturar y almacenar el conocimiento de expertos humanos en la base de conocimientos del sistema.
- 5.Mantenimiento y actualización: Se refiere a la tarea continua de revisar y modificar la base de conocimientos para reflejar cambios en el dominio, incorporar nuevos conocimientos y mejorar el rendimiento del sistema.
- 6.En síntesis, los sistemas expertos representan un avance significativo en la aplicación práctica de la inteligencia artificial, siendo herramientas valiosas para la resolución de problemas complejos en una amplia gama de áreas.

“SISTEMA EXPERTO”

Los sistemas expertos actuales, cuando se utilizan, suelen integrar capacidades de aprendizaje automático, como el machine o el deep learning. Esto les permite mejorar el rendimiento y sacar el máximo partido a la experiencia acumulada. Hace al sistema experto más experto.

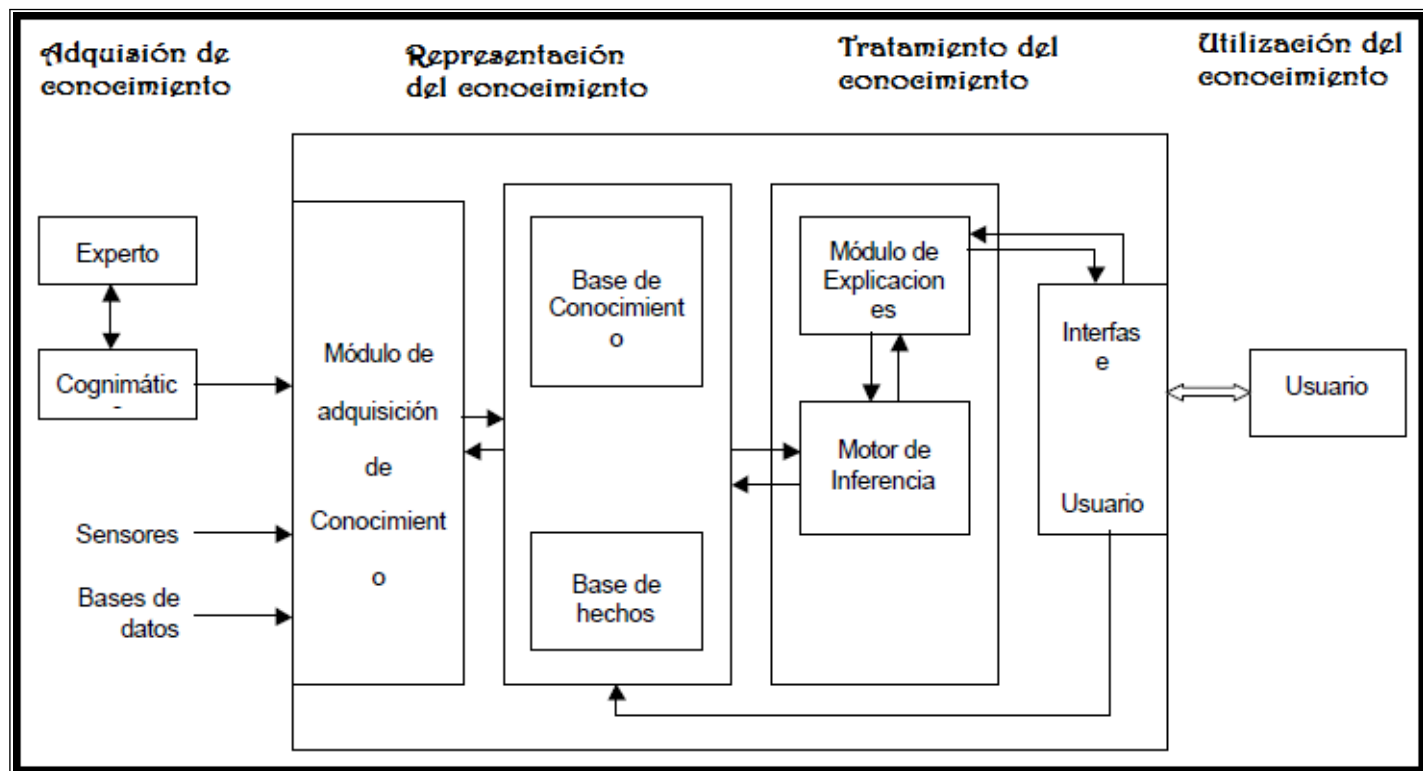
Entre las ventajas que ofrecen está la capacidad analítica y deductiva de una gran cantidad de información de forma muy rápida, lo que ahorra un tiempo muy valioso en la toma de decisiones que, de otro modo, habrían tenido que tomar seres humanos.

Es, por lo tanto, un ahorro de tiempo y recursos, pues estos sistemas una vez programados pueden replicarse infinitamente y usarse por todo el mundo, con la ventaja añadida de que no envejecen.

Sin embargo, en el paquete también se incluyen algunas desventajas considerables y que han provocado el desuso de muchos de estos sistemas por tecnologías con mejores prestaciones. Hablamos, por ejemplo, de la incapacidad para aprender que sí incluyen otros sistemas de IA, además de que para un sistema experto no existen deducciones más allá de las introducidas.

A continuación se muestran: La estructura de un Sistema experto está organizada alrededor de 3 elementos principales: Base de conocimiento, Motor de inferencia y Base de hechos. Pero para asegurar el diálogo entre el hombre y la máquina se requiere los siguientes:

Arquitectura de Un sistema Experto :



3. PROCEDIMIENTO

Uso de Sistema Experto: Clasificación de Héroes de Dota2

En el siguiente software empleando sistemas expertos, nos proporciona una interfaz gráfica simple para clasificar y agregar héroes de un videojuego, permitiendo al usuario explorar y gestionar la información de los héroes de manera interactiva.

CODIGO IMPLEMENTADO EN VISUAL STUDIO:

```
from ctypes import sizeof
from lib2to3.pgen2.token import LEFTSHIFT
from logging import RootLogger
from operator import length_hint
from select import select
from tkinter import *
from tkinter import filedialog as fd
import shutil
import copy
import os
import tkinter
from turtle import width
from PIL import ImageTk, Image
import numpy as np
import time
import matplotlib.pyplot as plt
import matplotlib
matplotlib.use('Agg')
import threading
import os
import random
matplotlib.use('TkAgg')
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg
from ctypes import sizeof
from lib2to3.pgen2.token import LEFTSHIFT
from logging import RootLogger
from operator import length_hint
from select import select
from tkinter import *
from tkinter import filedialog as fd
import shutil
import os
from tkinter import *
from tkinter import filedialog as fd
from PIL import ImageTk, Image

class Hero:
    def __init__(self):
        self.name = ""
        self.lore = ""
        self.primary_attribute = ""
        self.attack_type = ""
        self.image = ""

class Visualizer:
    def __init__(self, menu, frame1, hero):
        self.menu = menu
```

```

self.frame1 = frame1
self.hero = hero
self.name_label = Label(self.frame1, text="Heroe", background='#353437', fg="white")
self.name_label.configure(font=("Arial", 35))
open_image = Image.open(hero.image)
resized_image = open_image.resize((200, 200))
self.photo = ImageTk.PhotoImage(resized_image)
self.image_label = Label(self.frame1, image=self.photo)
self.lore_label = Label(self.frame1, text=hero.lore, wraplength=1200, background='#353437', fg="white")
self.lore_label.configure(font=("Arial", 14))
self.primary_attribute_label = Label(self.frame1, text=hero.primary_attribute, wraplength=1200, background='#353437', fg="white")
self.primary_attribute_label.configure(font=("Arial", 14))
self.attack_type_label = Label(self.frame1, text=hero.attack_type, wraplength=1200, background='#353437', fg="white")
self.attack_type_label.configure(font=("Arial", 14))
self.menu_button = Button(self.frame1, text="Menu Principal", command=self.return_to_menu, bg="black", fg="white")
self.menu_button.config(height=2, width=15)

def show(self):
    self.clear_frame()
    self.name_label.pack()
    self.image_label.pack()
    self.lore_label.pack()
    self.primary_attribute_label.pack()
    self.attack_type_label.pack()
    self.menu_button.pack()

def clear_frame(self):
    for widget in self.frame1.winfo_children():
        widget.pack_forget()

def return_to_menu(self):
    self.menu.show()

class NewHeroForm:
    def __init__(self, menu, frame1, classifier):
        self.menu = menu
        self.frame1 = frame1
        self.classifier = classifier

        self.name_label = Label(self.frame1, text="Nombre del Héroe:", background='#353437', fg="white")
        self.name_label.configure(font=("Arial", 20))
        self.name_entry = Entry(self.frame1)

        self.lore_label = Label(self.frame1, text="Historia del Héroe:", background='#353437', fg="white")
        self.lore_label.configure(font=("Arial", 20))
        self.lore_entry = Entry(self.frame1, width=50)

        self.attribute_label = Label(self.frame1, text="Atributo Primario:", background='#353437', fg="white")
        self.attribute_label.configure(font=("Arial", 20))
        self.attribute_options = ["Fuerza", "Agilidad", "Inteligencia", "Otros"]
        self.attribute_selection = StringVar(self.frame1)
        self.attribute_selection.set(self.attribute_options[0])
        self.attribute_dropdown = OptionMenu(self.frame1, self.attribute_selection, *self.attribute_options)

        self.attack_type_label = Label(self.frame1, text="Tipo de Ataque:", background='#353437', fg="white")
        self.attack_type_label.configure(font=("Arial", 20))
        self.attack_type_options = ["Cuerpo a cuerpo", "A distancia"]
        self.attack_type_selection = StringVar(self.frame1)
        self.attack_type_selection.set(self.attack_type_options[0])

```

```

self.attack_type_dropdown = OptionMenu(self.frame1, self.attack_type_selection, *self.attack_ty

self.image_label = Label(self.frame1, text="Ruta de la Imagen:", background='#353437', fg="whit
self.image_label.configure(font=("Arial", 20))
self.image_entry = Entry(self.frame1, width=50)
self.image_button = Button(self.frame1, text="Buscar Imagen", command=self.browse_image)

self.submit_button = Button(self.frame1, text="Agregar Héroe", command=self.add_hero, bg="#7a7b
self.menu_button = Button(self.frame1, text="Menu Principal", command=self.return_to_menu, bg="
                        fg="white")

def show(self):
    self.clear_frame()
    self.name_label.pack()
    self.name_entry.pack()
    self.lore_label.pack()
    self.lore_entry.pack()
    self.attribute_label.pack()
    self.attribute_dropdown.pack()
    self.attack_type_label.pack()
    self.attack_type_dropdown.pack()
    self.image_label.pack()
    self.image_entry.pack()
    self.image_button.pack()
    self.submit_button.pack()
    self.menu_button.pack()

def clear_frame(self):
    for widget in self.frame1.winfo_children():
        widget.pack_forget()

def browse_image(self):
    file_path = fd.askopenfilename(filetypes=[("Image Files", "*.jpg;*.png")])
    if file_path:
        self.image_entry.delete(0, END)
        self.image_entry.insert(0, file_path)

def add_hero(self):
    new_hero = Hero()
    new_hero.name = self.name_entry.get()
    new_hero.lore = self.lore_entry.get()
    new_hero.primary_attribute = self.attribute_selection.get()
    new_hero.attack_type = self.attack_type_selection.get()
    new_hero.image = self.image_entry.get()
    self.classifier.heroes.append(new_hero)
    self.return_to_menu()

def return_to_menu(self):
    self.menu.show()

class NotFoundScreen:
    def __init__(self, menu, frame1):
        self.menu = menu
        self.frame1 = frame1
        self.message_label = Label(self.frame1, text="Héroe no registrado.", background='#353437', fg="
        self.message_label.configure(font=("Arial", 30))
        self.add_button = Button(self.frame1, text="Agregar Nuevo Héroe", command=self.menu.show_add_he
                                bg="#7a7b7c", fg="white")
        self.menu_button = Button(self.frame1, text="Menu Principal", command=self.return_to_menu, bg="
                                fg="white")
        self.not_found_image_path = "sources/noexiste.jpg"

```

```

        self.not_found_image = Image.open(self.not_found_image_path)
        self.not_found_image = self.not_found_image.resize((200, 200))
        self.photo = ImageTk.PhotoImage(self.not_found_image)
        self.image_label = Label(self.frame1, image=self.photo)

    def show(self):
        self.clear_frame()
        self.message_label.pack()
        self.add_button.pack()
        self.menu_button.pack()
        self.image_label.pack()

    def clear_frame(self):
        for widget in self.frame1.winfo_children():
            widget.pack_forget()

    def return_to_menu(self):
        self.menu.show()

class Classifier:
    def __init__(self, menu, frame1):
        self.menu_window = menu
        self.frame1 = frame1
        self.heroes = []
        self.load_heroes()
        self.primary_attribute_selection = None
        self.attack_type_selection = None

    def get_unique_primary_attributes(self):
        attributes = set(hero.primary_attribute for hero in self.heroes)
        return list(attributes) + ["Otros"]

    def get_unique_attack_types(self):
        types = set(hero.attack_type for hero in self.heroes)
        return list(types)

    def classify(self):
        options = self.get_unique_primary_attributes()
        self.primary_attribute_selection = StringVar(self.frame1)
        self.primary_attribute_selection.set(options[0])
        self.primary_attribute_label = Label(self.frame1, text="Seleccione el Atributo Primario:", background="white", font=("Arial", 25))
        self.primary_attribute_label.pack()
        self.drop = OptionMenu(self.frame1, self.primary_attribute_selection, *options)
        self.drop.config(height=1, width=20)
        self.drop.pack()
        self.button = Button(self.frame1, text="Next", command=self.show_next_screen, background="#7a7b7c", foreground="white", font=("Arial", 12))
        self.button.config(height=2, width=10)
        self.button.pack()

    def show_next_screen(self):
        selected_attribute = self.primary_attribute_selection.get()
        if selected_attribute == "Otros":
            self.show_not_found_screen()
        else:
            self.show_attack_type_options()

    def show_attack_type_options(self):
        options = self.get_unique_attack_types()
        self.attack_type_selection = StringVar(self.frame1)
        self.attack_type_selection.set(options[0])

```

```

self.attack_type_label = Label(self.frame1, text="Seleccione el Tipo de Ataque:", background='#f0f0f0')
self.attack_type_label.configure(font=("Arial", 25))
self.attack_type_label.pack()
self.drop = OptionMenu(self.frame1, self.attack_type_selection, *options)
self.drop.config(height=1, width=20)
self.drop.pack()
self.button = Button(self.frame1, text="Next", command=self.filter_heroes, bg="#7a7b7c", fg="white")
self.button.config(height=2, width=10)
self.button.pack()

def filter_heroes(self):
    selected_attribute = self.primary_attribute_selection.get()
    selected_attack_type = self.attack_type_selection.get()
    self.possible_heroes = [hero for hero in self.heroes if
                            hero.primary_attribute == selected_attribute and hero.attack_type == selected_attack_type]
    if self.possible_heroes:
        self.visualize_heroes()
    else:
        self.show_not_found_screen()

def visualize_heroes(self):
    if self.possible_heroes:
        self.visual = Visualizer(self.menu_window, self.frame1, self.possible_heroes[0])
        self.visual.show()

def show_not_found_screen(self):
    self.not_found = NotFoundScreen(self.menu_window, self.frame1)
    self.not_found.show()

def load_heroes(self):
    # Pudge
    pudge = Hero()
    pudge.name = "Pudge"
    pudge.lore = "Pudge, el Carnicero, es un héroe cuerpo a cuerpo conocido por su habilidad de gan"
    pudge.primary_attribute = "Fuerza"
    pudge.attack_type = "Cuerpo a cuerpo"
    pudge.image = "sources/pudge.jpg"
    self.heroes.append(pudge)

    # Invoker
    invoker = Hero()
    invoker.name = "Invoker"
    invoker.lore = "Invoker, el Invocador, es un héroe de inteligencia conocido por su habilidad pa"
    invoker.primary_attribute = "Inteligencia"
    invoker.attack_type = "A distancia"
    invoker.image = "sources/invoker.jpg"
    self.heroes.append(invoker)

    # Sven
    sven = Hero()
    sven.name = "Huskar"
    sven.lore = "Huskar, el Sangre Ardiente, es un héroe de fuerza conocido por su capacidad de ata"
    sven.primary_attribute = "Fuerza"
    sven.attack_type = "A distancia"
    sven.image = "sources/huskar.jpg"
    self.heroes.append(sven)

    # Void
    void = Hero()
    void.name = "Void"
    void.lore = "Faceless Void, el Vacío, es un héroe de agilidad conocido por su habilidad de mani"

```



```

void.primary_attribute = "Agilidad"
void.attack_type = "Cuerpo a cuerpo"
void.image = "sources/void.jpg"
self.heroes.append(void)

# Windranger
windranger = Hero()
windranger.name = "Windranger"
windranger.lore = "Windranger, la Ranger, es una heroína de agilidad conocida por su gran movilidad"
windranger.primary_attribute = "Agilidad"
windranger.attack_type = "A distancia"
windranger.image = "sources/windranger.jpg"
self.heroes.append(windranger)

class MainMenu:

    def __init__(self, root):
        self.root = root
        self.frame1 = Frame(root, background='#353437')
        self.title = Label(self.frame1, text="Clasificador de Héroes\n\n\n", font=("Arial", 25), background=
            fg="white")
        self.classifier_button = Button(self.frame1, text="Encontrar Héroe", command=self.show_classifier,
            bg="#7a7b7c", fg="white")
        self.classifier_button.config(height=5, width=30)
        self.classifier_window = Classifier(self, self.frame1)
        self.new_hero_form = NewHeroForm(self, self.frame1, self.classifier_window)

    def show(self):
        self.frame1.pack(pady=20)
        self.title.pack()
        self.classifier_button.pack()

    def hide(self):
        self.title.pack_forget()
        self.classifier_button.pack_forget()
    def show_classifier_window(self):
        self.hide()
        self.classifier_window.classify()
    def show_add_hero_form(self):
        self.hide()
        self.new_hero_form.show()
    def closing(self):
        self.root.destroy()

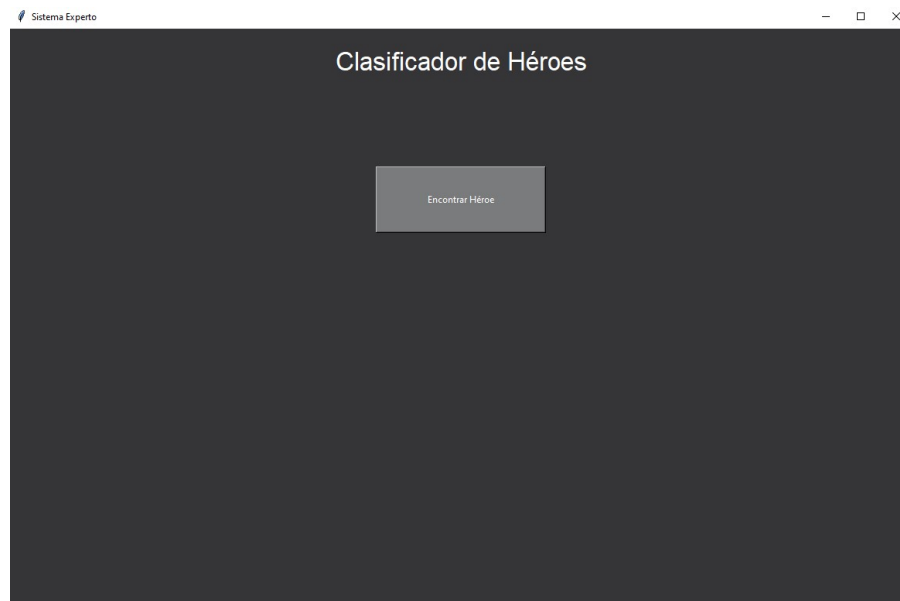
if __name__ == "__main__":
    try:
        root = Tk()
        root.configure(bg='#353437')
        root.title("Sistema Experto")
        w, h = root.winfo_screenwidth(), root.winfo_screenheight()
        root.geometry("%dx%d" % (w, h))

        program = MainMenu(root)
        def on_closing():
            program.closing()
        root.protocol("WM_DELETE_WINDOW", on_closing)
        program.show()
        root.mainloop()
    except Exception as e:
        print(e)

```

EJECUCION DEL CODIGO:

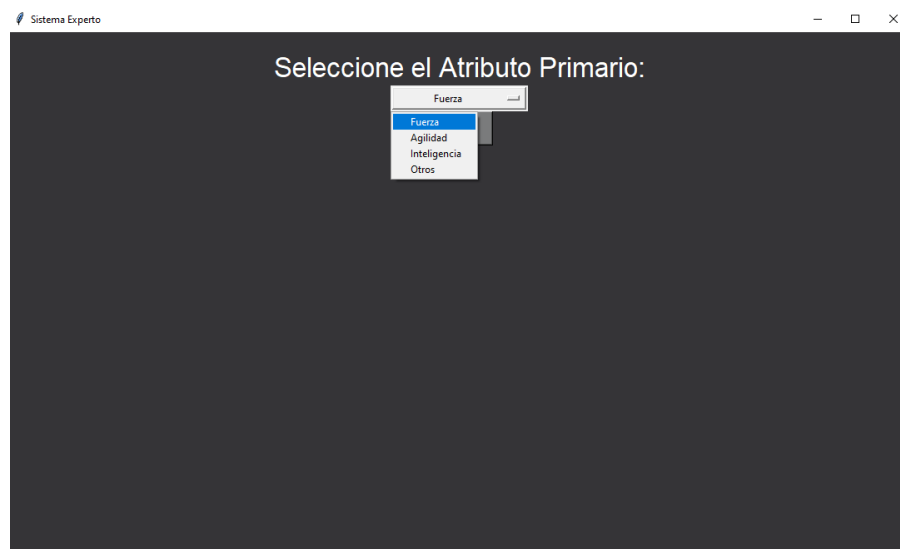
1.



Analysis:

El clasificador de héroes brindara a los usuarios una manera fácil y rápida de buscar y explorar información sobre los héroes del videojuego, lo que facilita la gestión y el conocimiento de los personajes disponibles en el juego.

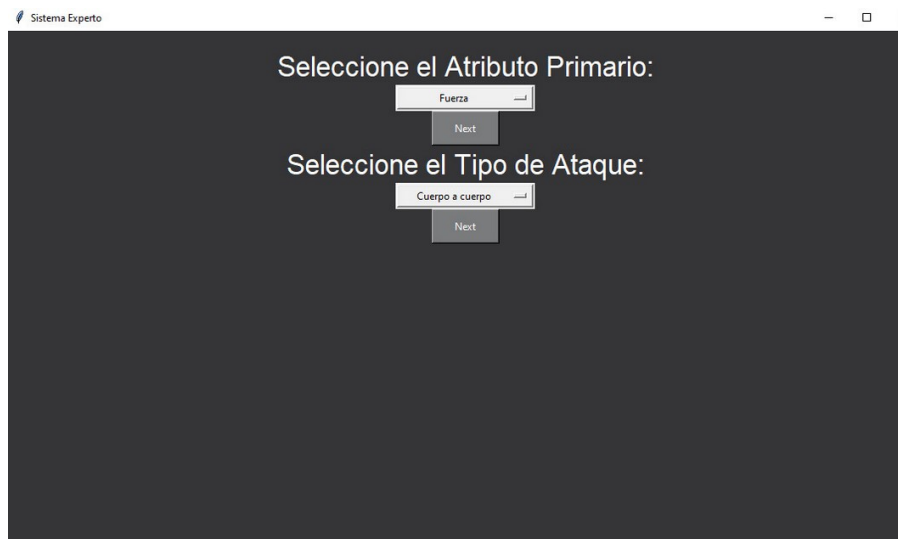
2.



Analysis:

En esta sección vemos que podemos elegir el atributo primario proporciona una experiencia de usuario intuitiva y eficiente, permitiendo al usuario seleccionar un criterio importante para la búsqueda de héroes de manera clara y concisa.

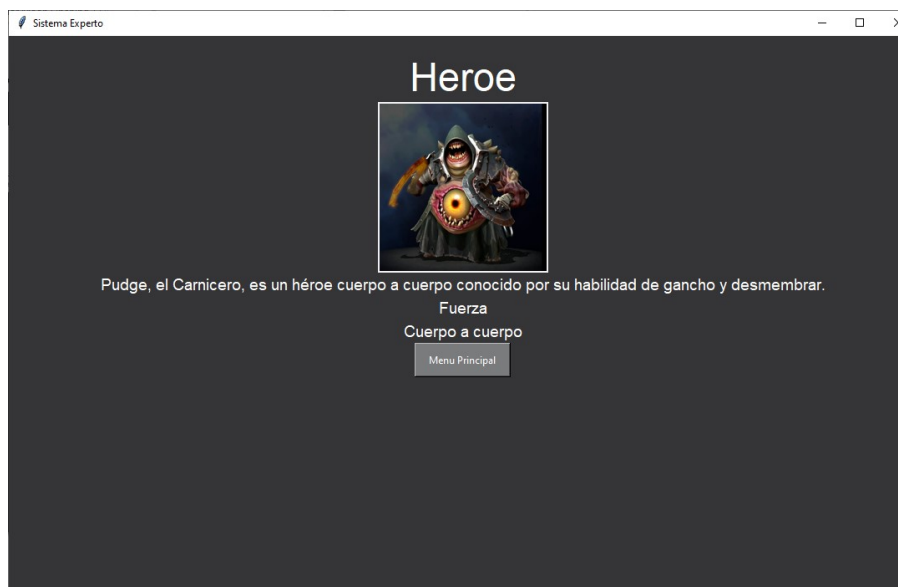
3.



Analisis:

La selección del tipo de ataque ofrece una interfaz simplificada y fácil de entender para que el usuario elija el estilo de combate deseado del héroe,

4.



Analisis:

Finalmente después de seleccionar "Fuerza" como el atributo primario y "Cuerpo a cuerpo" como el tipo de ataque, el programa filtra la lista de héroes y presenta los resultados correspondientes al usuario para su exploración y consulta, en este caso nos muestra al Heroe Pudge, así como una breve descripción del héroe y el tipo de ataque y su atributo primario.

5. Seleccionaremos en esta Ocasión el mismo atributo pero otro Tipo de Ataque:

Sistema Experto

Seleccione el Atributo Primario:

Fuerza


Next

Seleccione el Tipo de Ataque:

A distancia

Next

Heroe



Huskar, el Sangre Ardiente, es un héroe de fuerza conocido por su capacidad de ataque a distancia y su agresividad en el juego temprano.

Fuerza

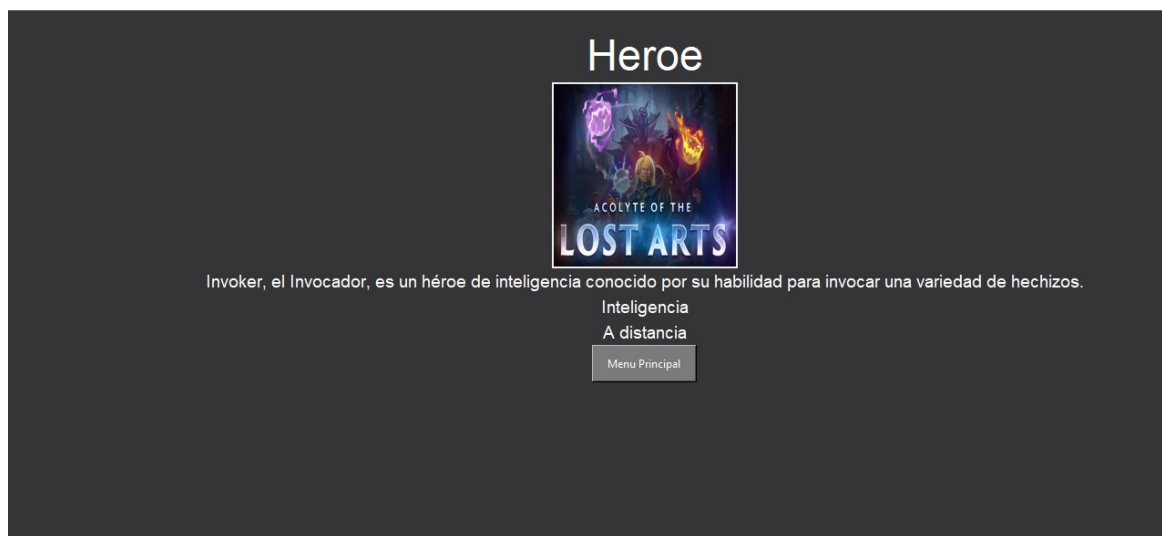
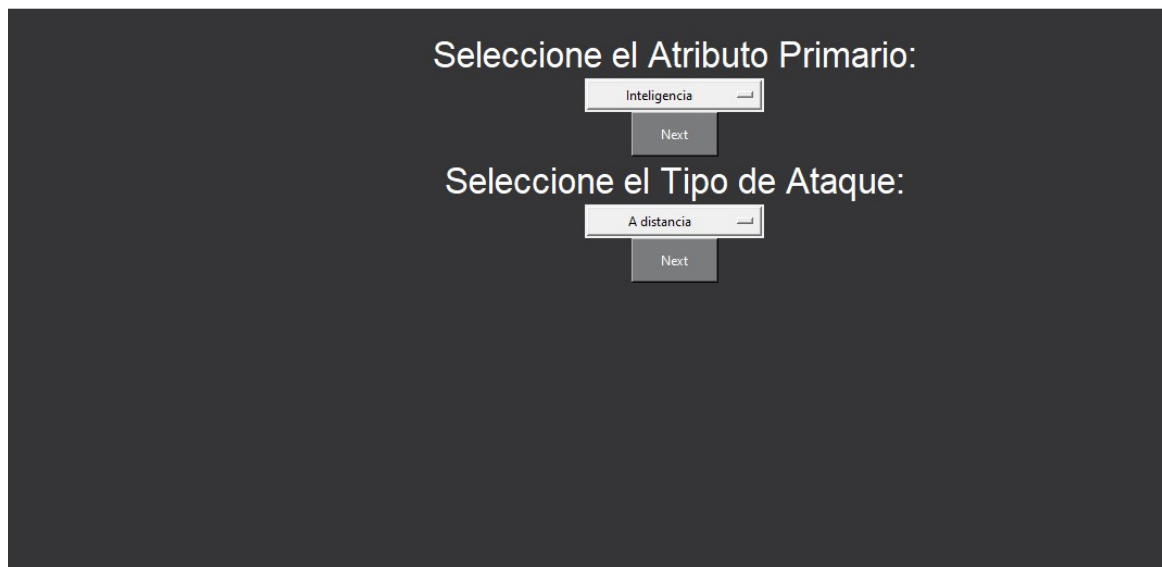
A distancia

Menu Principal

Analisis:

Después de seleccionar "Fuerza", como el atributo primario y con el nuevo cambio. A Distancia, como el tipo de ataque, el programa filtra la lista de héroes y presenta los resultados correspondientes al usuario para su exploración y consulta, en este caso nos muestra al Heroe Huskar, así como una breve descripción del héroe y el tipo de ataque y su atributo primario.

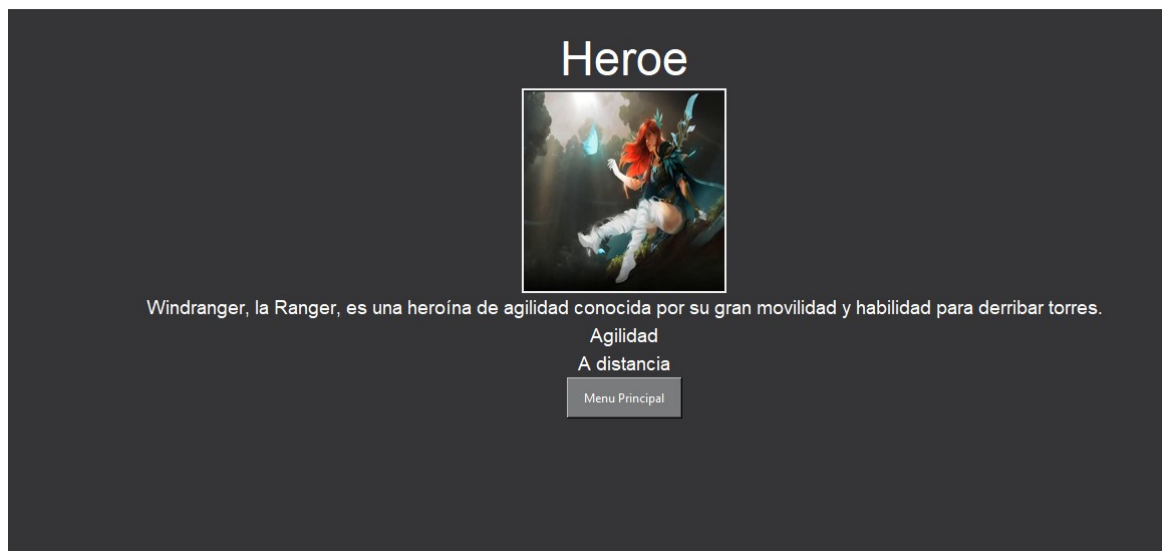
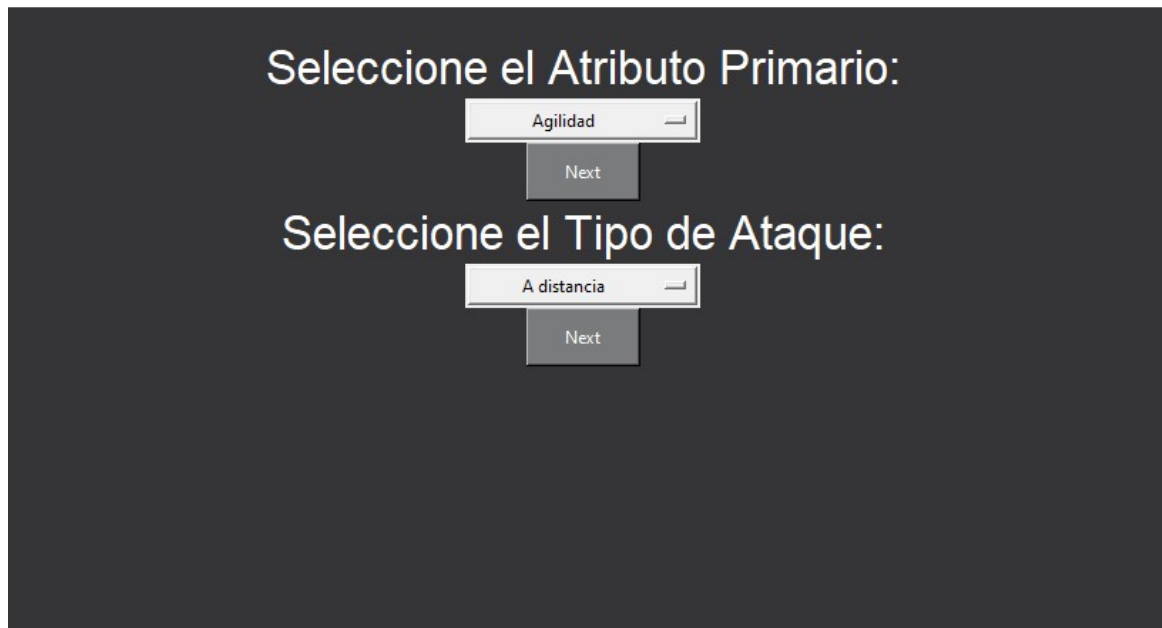
6. Seleccionamos otro Atributo Primario para Probar mas Heroes:



Analisis:

Después de seleccionar "Inteligencia", como el atributo primario y con seleccionamos . A Distancia, como el tipo de ataque, el programa filtra la lista de héroes y presenta los resultados correspondientes al usuario para su exploración y consulta, en este caso nos muestra al Heroe Invoker, así como una breve descripción del héroe y el tipo de ataque y su atributo primario.

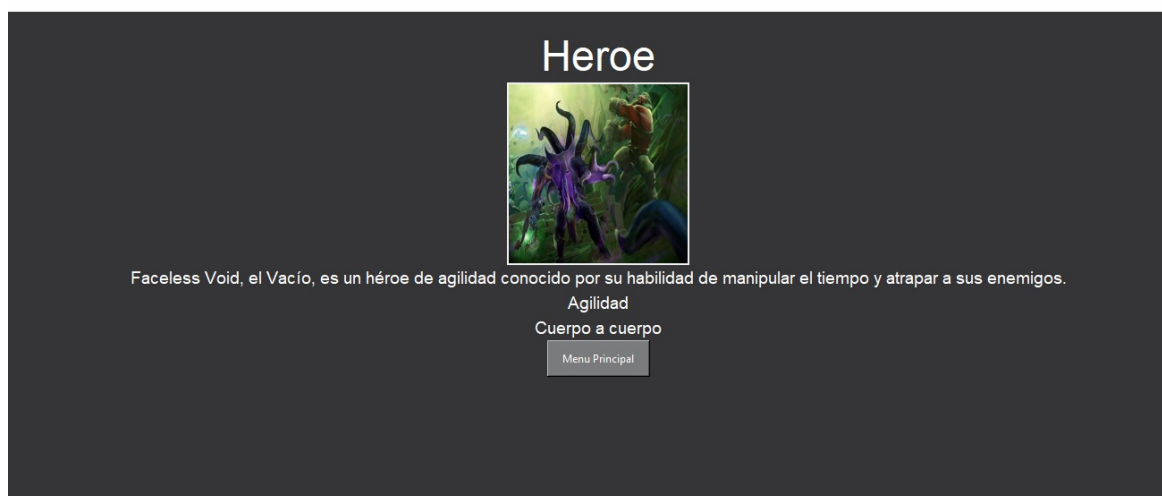
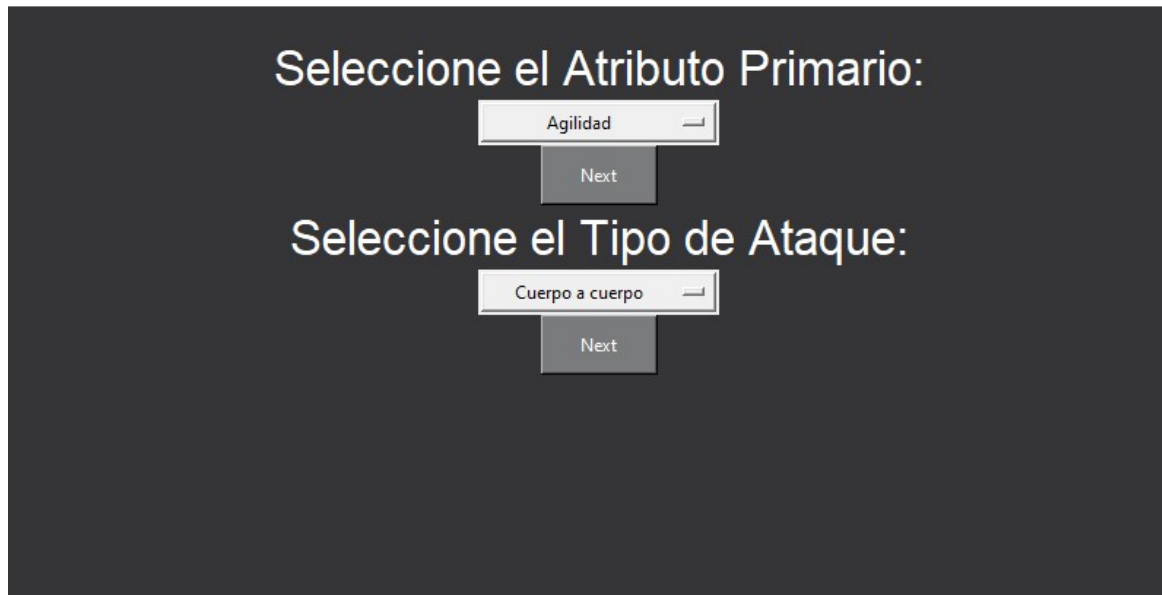
7. Seleccionamos otro Atributo Primario:



Analysis:

Seleccionar "Agilidad", como el atributo primario y seleccionamos, "A Distancia", como el tipo de ataque, en este caso nos muestra al Heroe Windranger, así como una breve descripción del hero y el tipo de ataque y su atributo primario.

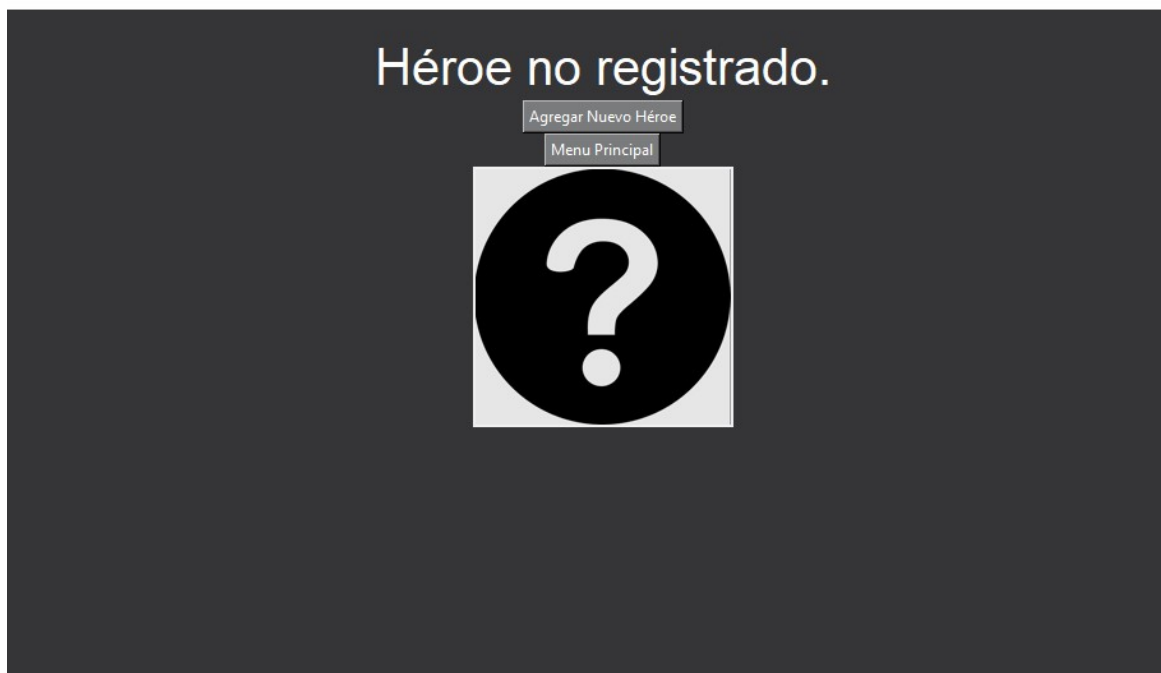
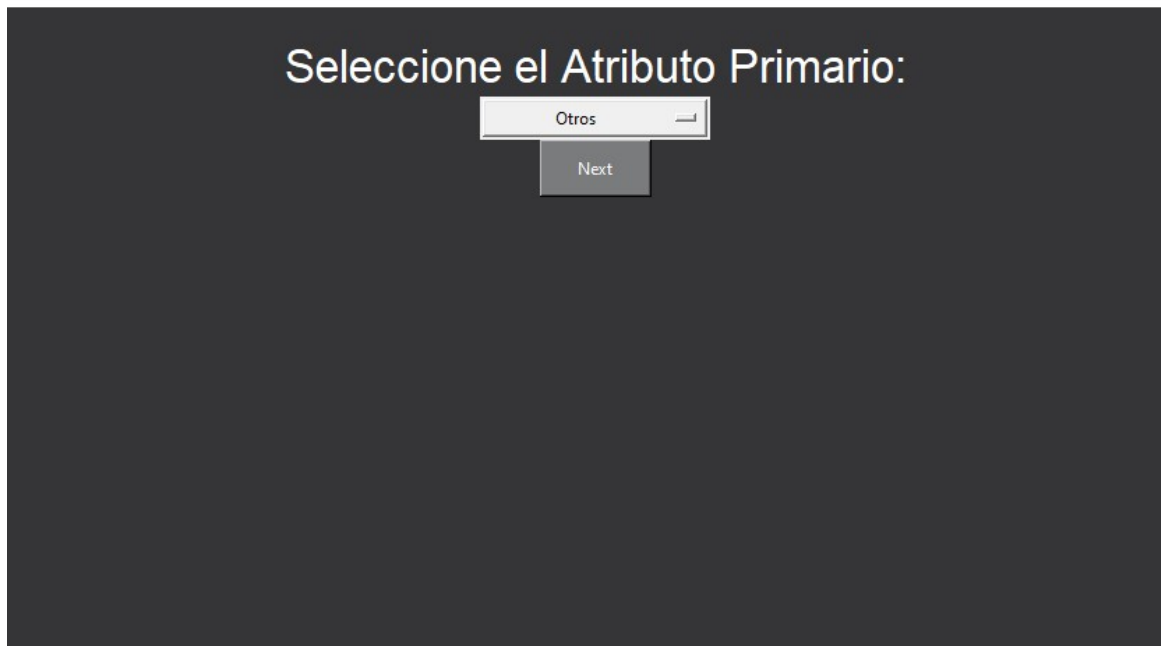
8. Seleccionamos el mismo Atribuo Primerio pero distinto tipo de ataque:



Analisis:

Seleccionamos "Agilidad", como el atributo primario y seleccionamos, "Cuerpo a Cuerpo", como el tipo de ataque, en este caso nos muestra al Heroe Faceless Void, así como una breve descripción del hero y el tipo de ataque y su atributo primario.

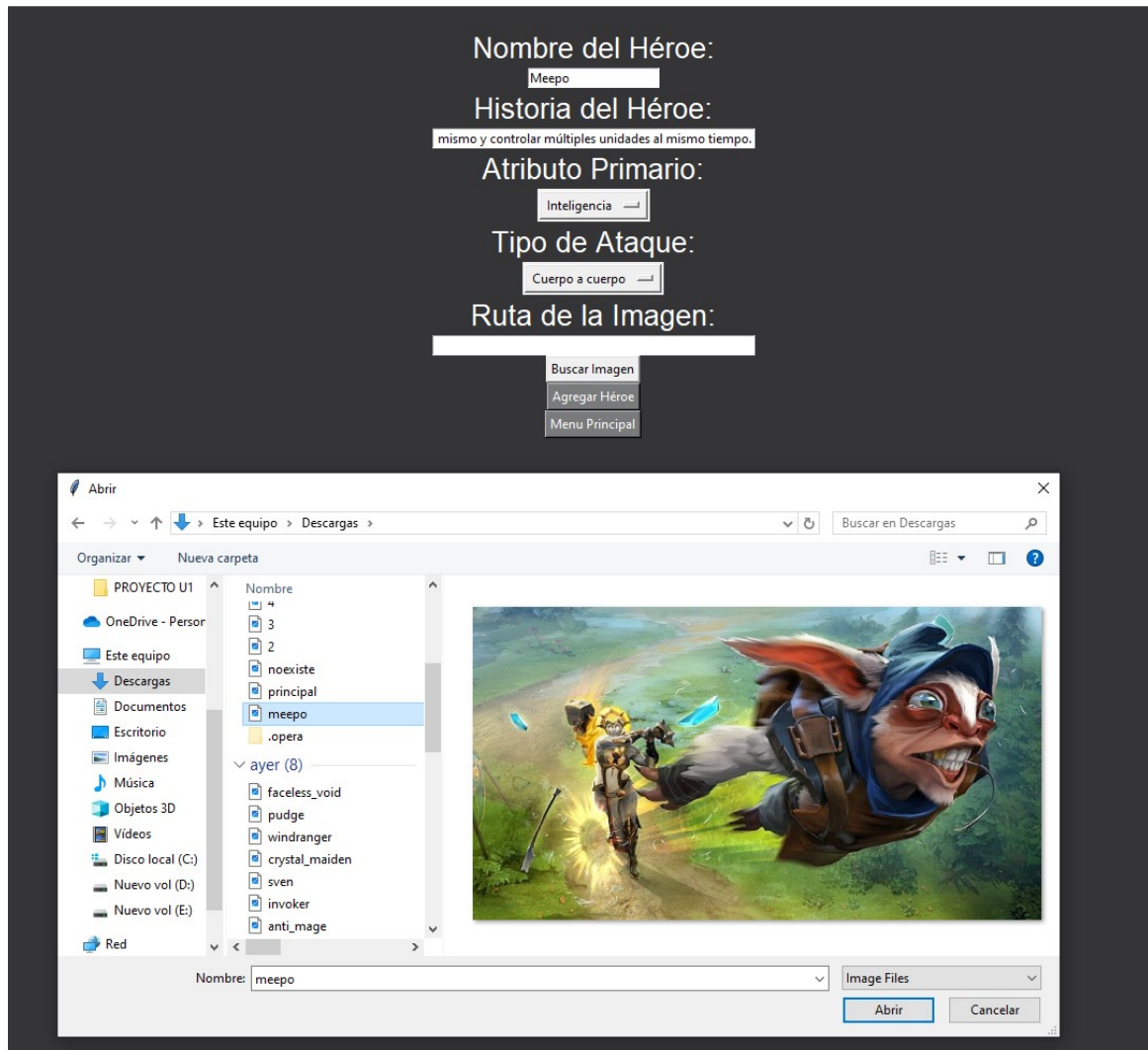
9.En esta parte Seleccionaremos el atributo Otros y veremos un mensaje de "Héroe no registrado":



Analisis:

Al Seleccionar en el atributo Principal la opcion,Otros, veremos una interfaz que nos da la opcion y el mensaje que el Héroe no se encuentra registrado,por lo tanto procederemos a hacer clic en .Agregar Nuevo Héroe",y procederemos a registrarlo a continuación.

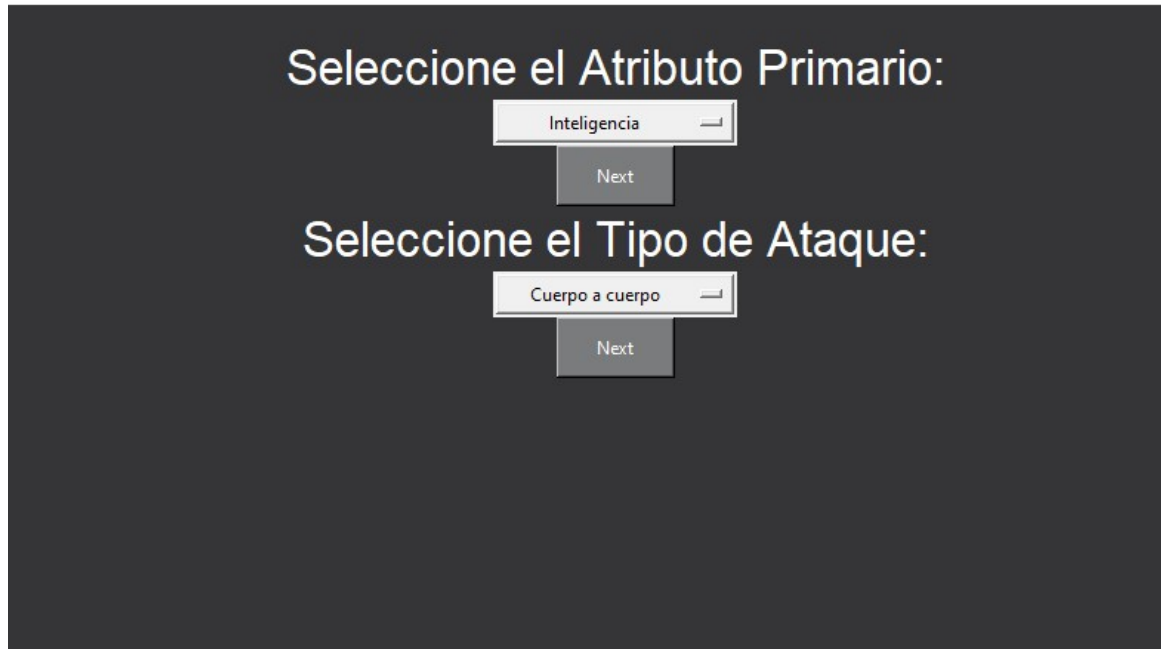
10. Apreciamos la Interfaz con los datos a agregar del nuevo Heroe, el nombre, la historia, su atributo Primario y el tipo de ataque, así como la ruta de imagen a agregar y daremos clic en "Agregar Heroe".



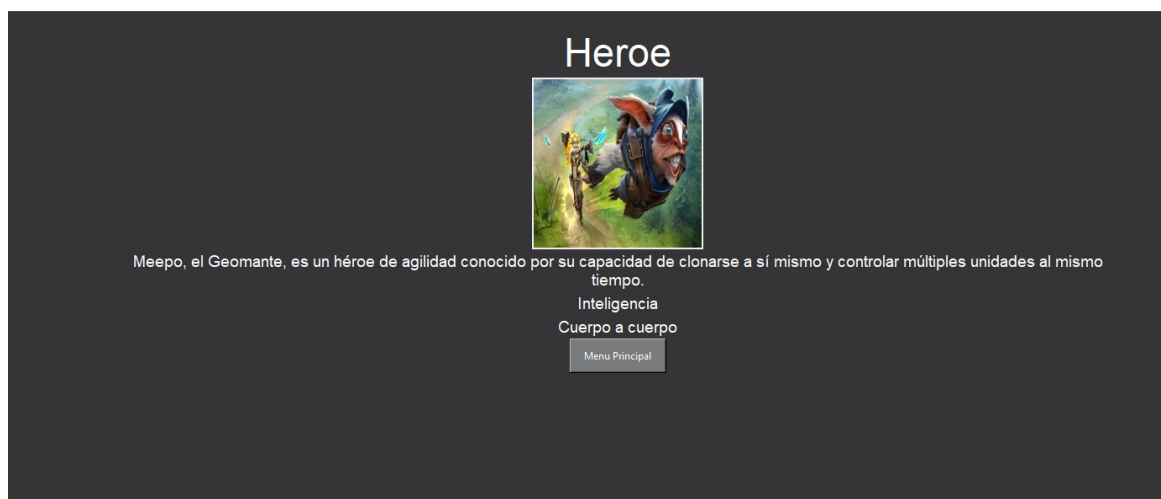
Analisis:

Al Seleccionar en el atributo Principal la opcion Otros, veremos una interfaz que nos da la opcion y el mensaje que el Heroe no se encuentra registrado, por lo tanto procederemos a hacer clic en Agregar Nuevo Heroe, y procederemos a registrarlo a continuación.

11.Una vez Registrado el Heroe procedemos a buscarlo nuevamente,seleccionaremos El atributo y el Tipo de ataque que fue Registrado en este caso Cuerpo a Cuerpo e Inteligencia como Atributo Primario:.



12.Apreciamos el Heroe que fue Registrado con su atributo Primario y el Tipo de ataque,asi como su nombre y descripcion:



Analisis:

Al haber registrado a nuestro nuevo heroe ,apreciamos que se registro exitosamente y asi podremos seguir registrando nuevos Heroes.

Explicacion:

El código proporcionado es un ejemplo de una aplicación que se encarga de clasificar héroes ficticios del juego Dota 2, según ciertas características como su atributo principal y su tipo de ataque. Está estructurado en varias clases, cada una con funciones específicas. Por ejemplo, la clase Hero define las propiedades de cada héroe, como su nombre y atributos. La clase Visualizer se encarga de mostrar la información de un héroe en la interfaz de usuario, mientras que la clase NewHeroForm permite agregar nuevos héroes al sistema a través de un formulario interactivo.

Finalmente todo ello nos ayuda a aprender e implementar un sistema experto básico para la clasificación de héroes del juego Dota 2.

4. Conclusiones

1. Como conclusiones, una de las ventajas de los sistemas expertos ofrecen una herramienta invaluable para emular el conocimiento y la experiencia de expertos humanos en diversos campos. Su capacidad para analizar grandes cantidades de datos, interpretarlos y ofrecer recomendaciones precisas los convierte en activos fundamentales para la toma de decisiones y la resolución de problemas complejos, que es el objetivo del curso.
2. Durante el desarrollo del trabajo, encuentran aplicaciones en una amplia gama de áreas, desde el análisis financiero y la medicina hasta la ingeniería y la informática. Su flexibilidad y adaptabilidad los hacen útiles en situaciones donde se requiere un alto grado de conocimiento especializado y consistencia en la toma de decisiones.
3. Aunque los sistemas expertos ofrecen numerosos beneficios, también enfrentan desafíos y limitaciones. La adquisición y actualización de conocimientos, la precisión de las recomendaciones y la interpretación de información ambigua son áreas que requieren atención continua para mejorar la efectividad y la confiabilidad de estos sistemas.

Enlace del Proyecto en el GitHub:

<https://github.com/AntonyChata/Sistemas-Expertos—Heroes-Dota>