Selección del Programa y Diseño de Diagramas Funcionales y Arquitectura

Asignatura: Lógica de Programación
Alumno: Jonathan José Pedraza Pilataxi
Fecha: 24 de agosto de 2025

1. Investigar los tipos de Diagramas de funcionalidad y arquitectura de aplicaciones que existen y seleccionar uno de cada uno.

1.1 Diagramas de Funcionalidad

Es una herramienta visual que permite representar cómo funciona un sistema. Ayudan a visualizar los flujos del proceso, las interacciones y las relaciones que existen entre los diversos componentes del sistema. (Carter, 2024)

- **Diagrama de Casos de Uso (UML):** Muestra la relación entre los usuarios y las funciones del sistema. Es útil para entender los requisitos de alto nivel.
- **Diagrama de Actividades (UML):** Representa el flujo de control de una actividad a otra, similar a un diagrama de flujo. Es útil para modelar procesos de negocio y flujos de trabajo.
- **Diagrama de Secuencia (UML):** Ilustra cómo los objetos interactúan entre sí en una secuencia de tiempo para lograr una tarea. Es útil para mostrar la dinámica de las interacciones entre componentes.
- **Diagrama de Flujo de Datos (DFD):** Muestra el flujo de información a través de un sistema. Es útil para visualizar la entrada, el procesamiento y la salida de los datos.

1.2 Diagramas de Arquitectura.

Los diagramas de arquitectura se centran en la estructura del software, sus componentes, las relaciones entre ellos y cómo se implementarán.

- **Diagrama de Componentes**: Muestra la organización y piensa en los componentes como grupo de clases. Es útil para modelar la estructura del sistema a gran escala.
- **Diagrama de Despliegue**: Representa la disposición física de los nodos (hardware) y el despliegue de los artefactos de software en esos nodos. Es ideal para mostrar cómo se distribuirá la aplicación en diferentes entornos.

- **Diagrama de estados**: Muestra como un objeto transita entre diferentes estados, del proceso, es útil para modelar el comportamiento del sistema ante diferentes eventos.
- **Diagrama de Arquitectura de Capas**: Organiza el sistema en capas horizontales, donde cada capa tiene una responsabilidad específica. Es muy común en el desarrollo de software, ya que tiene presentación, lógica de negocio, acceso a datos. (Hielo, 2024)

JUEGO: **El juego del ahorcado.**

Se trabajará con el **diagrama de casos de uso**, ya que permite identificar las funciones principales del juego, es decir, iniciar le juego, adivinar la letra y ver estado del usuario.

En cuanto a la arquitectura se usará el **diagrama de arquitectura de capas**, ya que nos permite separar la lógica del juego de la interfaz del usuario, es decir:

Capa de presentación: Es la interfaz grafica del usuario que permite interactuar al jugador.

Capa de lógica de Negocio: Selección de palabras, adivinar, letras, verificación y estados.

Capa de datos: Aquí se almacenará las palabras y el estado del juego.

2. **Escoger el software a desarrollar y en función de eso realizar los pasos de resolución de problemas, (no se va a desarrollar aún el software pero si se deben entender muy bien que es lo que se va a resolver dentro de este problema de programación).**

JUEGO DEL AHORCADO.

Comprensión del problema.

El objetivo principal es crear un juego que sea capaz de adivinar la palabra, letra por letra con un número limitado de intentos por jugador.

- Entradas
    - Opción de Iniciar juego
    - Letras que el jugador puede ingresar en cada intento
    - Lista de palabras que el juego puede seleccionar
- Proceso
    - El sistema debe seleccionar una palabra al azar de una base de datos
    - Llevar un registro de letras acertadas correctamente y las incorrectas.
    - Contador de intentos fallidos para determinar el estado del ahorcado.
    - El juego termina si la palabra es adivinada por el jugador o llega al límite de intentos fallidos.
- Salidas
    - Estado actual de la palabra (letras adivinadas)
    - Letras que han sido utilizadas.

- Numero de intentos restantes.
- Mensajes de alerta y mensaje final (ganó o perdió).

Planificación de solución.

- Almacenamiento de Datos
  - Estructura como array o lista para guardar la lista de palabras.
  - Variables para guardar la palabra secreta actual y la palabra que verá el jugador (con guiones bajos)
  - Un array para guardar las letras adivinadas y un array para letras incorrectas.

- Flujo del juego:
  - Selecciona una palabra aleatoriamente
  - Inicia contador de intentos
  - Inicia palabra mostrada con guiones
  - Limpia la lista de letras usadas
  - Mostar el estado inicial del ahorcado.
  - Bucle:
    1. El jugador ingresa una letra.

    2. **Validar la entrada:** ¿Es una sola letra? ¿Ya se usó antes? ¿Es un número?

    3. V**erificar la suposición:**

       3.1 Si la letra está en la palabra secreta:
       Reemplazar los guiones en la palabra mostrada con la letra correcta.
       Informar al jugador que es correcto.

       3.2 Si la letra no está en la palabra secreta:
       Incrementar el contador de intentos fallidos.
       Informar al jugador que es incorrecto y mostrar letra usada.

    4. **Verificar condiciones de fin de juego:**
       Si la palabra mostrada es igual a la palabra secreta, el jugador gana.
       Si el contador de intentos fallidos llega al límite, el jugador pierde.

    5. Si el juego no ha terminado, se repite el bucle.

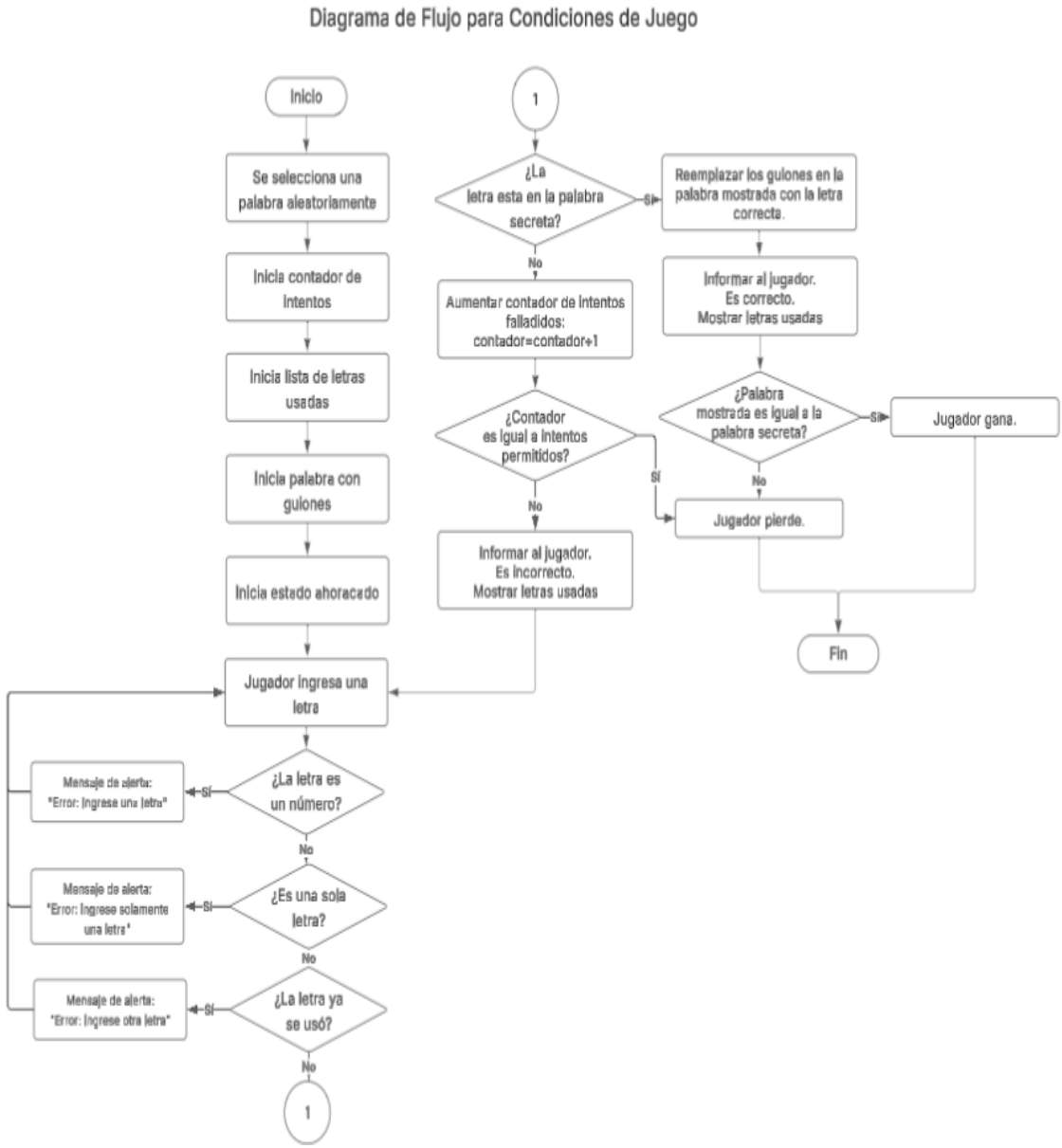
    6. Mostrar resultado final (ganador o perdedor) y palabra secreta.

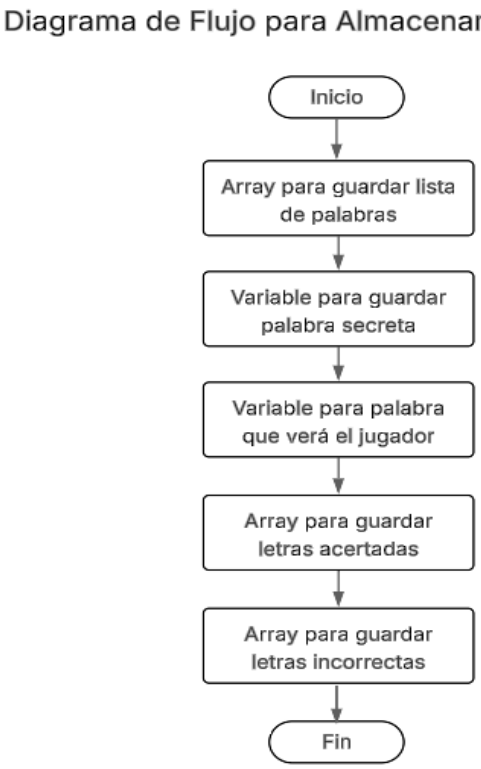
Se selecciona el Software Python ya que es ideal por su **simplicidad y legibilidad**. Nos permitirá enfocarnos en la lógica del juego sin complicaciones de sintaxis.

Además, cuenta con bibliotecas que facilitan el desarrollo, siendo un lenguaje muy común en proyectos.

Toda la programación se realizará en Visual Studio Code, ya que es un software versátil, eficiente y cuenta con todas las herramientas que se necesita para el proyecto, desde la escritura del código hasta la gestión con GitHub, además de ser gratuito y contar con múltiples lenguajes de programación.

3. **Fase de diseño de las funcionalidades, en esta fase se deben diseñar a detalle todo lo que va a ser capaz de hacer el software en función de diferentes diagramas.**





4. **Diseñar el diagrama de arquitectura de la aplicación en el que se muestra a un nivel macro como va a trabajar el software.**

Capa de presentación	Capa de lógica de Negocio	Capa de datos
Esta etapa comprende la interfaz grafica del usuario que permite interactuar al jugador. Aqui se verá el estado del ahorcado, asi como la palabra y mensajes de alerta.	En esta etapa, se realiza lo que es la selección de palabras, adivinar, verificación de letras tanto correctas como incorrectas, y el estado del juego.	Esta etapa permitirá almacenar las palabras y el estado actual del juego, se prevee lograr dibujar el ahorcado, mendiante el uso de funciones condicionales.

BIBLIOGRAFÍA

Carter, M. (24 de 09 de 2024). *Diagrama de bloques funcional: Qué es, tipos y cómo crearlo [+plantilla]*. Obtenido de boardmix: https://boardmix.com/es/knowledge/functional-block-diagram/

Hielo, P. d. (5 de 11 de 2024). *Los 7 tipos de diagramas UML más comunes para la arquitectura de software*. Obtenido de Medium: https://icepanel.medium.com/top-7-most-common-uml-diagram-types-for-software-architecture-3a43caaa2862