SECUENCIA DE UNA PARTIDA

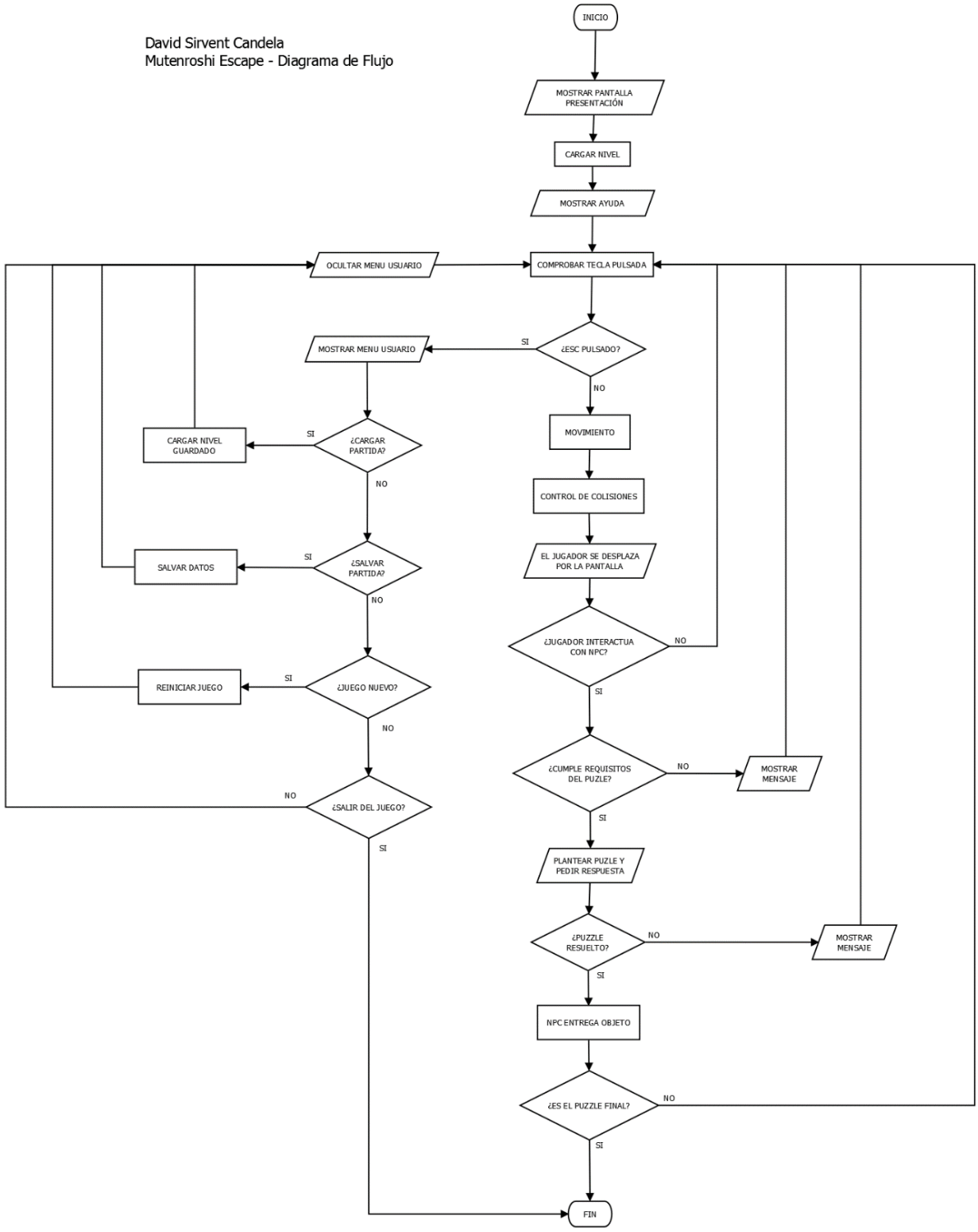
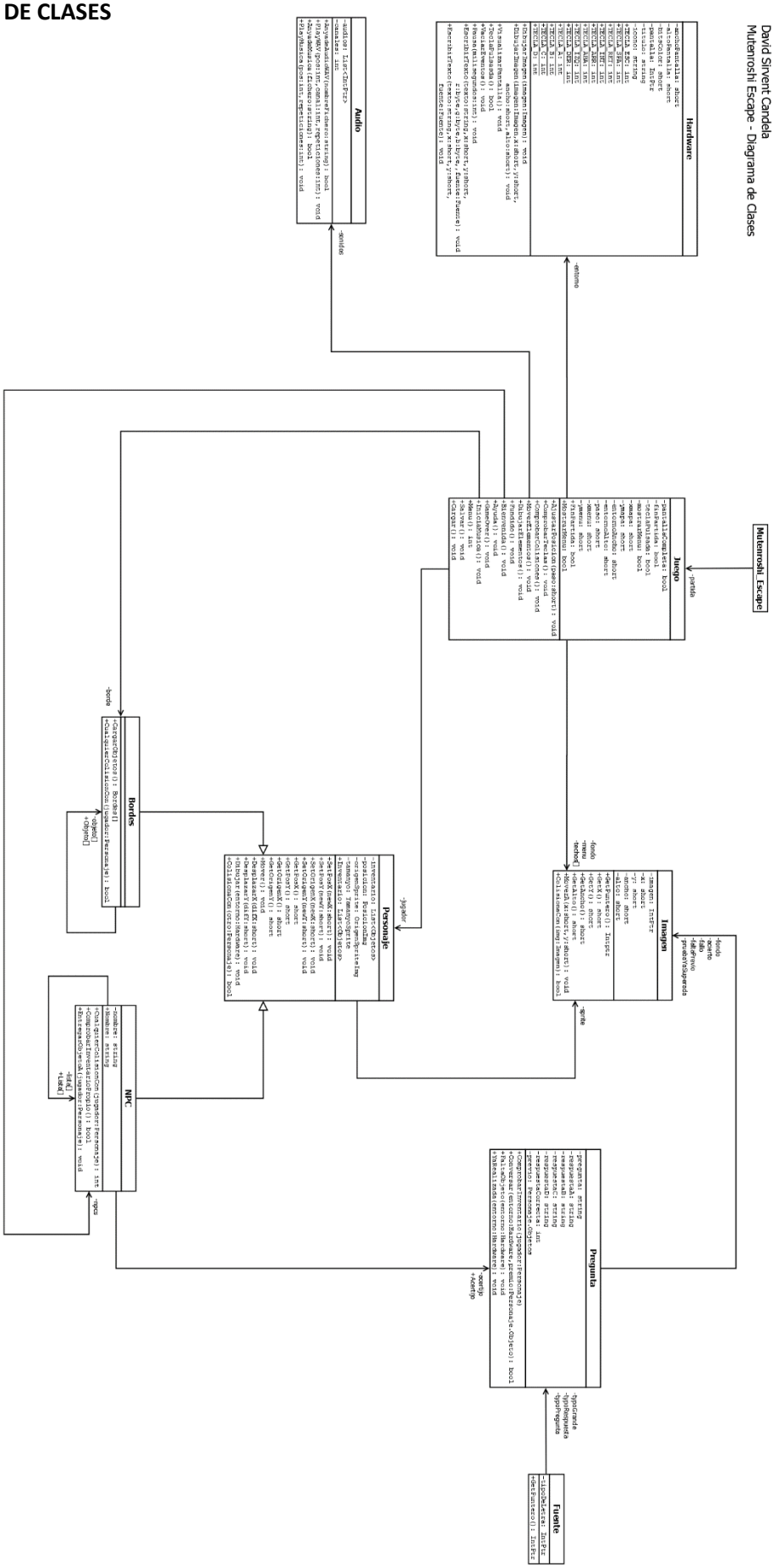


DIAGRAMA DE CLASES



DESCRIPCIÓN GENERAL DE OBJETOS Y MÉTODOS

- En el código existen comentarios que detallan las diferentes clases, atributos y métodos. En este texto se hace una presentación general y se remarcan o explican aquellas clases o métodos que puedan resultar más confusos.

- **Librería auxiliar Tao.Sdl**
  - Como base se utiliza Tao para preparar un entorno gráfico. Dicho entorno es gestionado por las siguientes clases:
    - **Hardware:** Maneja la renderización de imágenes y textos así como los eventos de entrada como teclado, ratón, gamepad, etc…
    - **Imagen:** Sirve para para incluir imágenes. Contiene el método *ColisionaCon()* que comprueba si un objeto está superpuesto a otro dado.
    - **Audio:** Ídem del anterior pero para audios.
    - **Fuentes:** Ídem de los anteriores pero para textos**.**
  - No es objeto de este manual profundizar en el uso del framework Tao.SDL.

- **Clase Juego**
  - Bajo esta clase se desarrolla todo el juego. Aquí se albergan las clases que manejan el desarrollo del juego como ComprobarTeclas(), MoverElementos(), DibujarElementos(), Menu(), Salvar(), Cargar().
  - Todas son bastantes autoexplicativas pero entraremos en detalle con Menu(), Salvar() y Cargar() y Ayuda().
    - Menu(): Dibuja una ventana superpuesta a todo que ofrece 4 opciones:
      - Juego nuevo: Crea una nueva instancia de Juego.
      - Salvar: ver método más abajo.
      - Cargar: ver método más abajo.
      - Salir: Abandonar juego.
    - Salvar(): Genera el fichero de texto “partida.dat” donde almaceno los datos necesarios de cada objeto del juego. Uno por línea identificando en el primer valor de la línea que tipo de objeto es. Sólo permito guardar un estado del juego.
      En un principio barajaba serializar toda la clase Juego, pero se producían una serie de errores (relacionados sobretodo con eventos) que no pude solucionar.
    - Cargar(): Si existe, lee el fichero “partida.dat” línea a línea y asigna a los diferentes objetos los valores guardados para un estado anterior del juego.
    - Ayuda(): Al inicio del juego se muestra una pantalla explicando el argumento del juego y los controles que se usan.

- **Clase Personaje**
  - Esta clase puede definir cualquier elemento que se posicione sobre el mapa. El propio jugador es una instancia directa de la clase y los NPCS, las paredes y objetos se instancian a partir de clases derivadas de esta.
  - Aparte de los atributos básicos de definen posición y tamaño tiene un inventario con capacidad para almacenar objetos del tipo enumerado “Personaje.Objetos” en una lista. Mientras que el jugador va acumulando objetos los npcs sólo poseen uno.

- **Clase NPC**
    - Esta clase define los personajes que interactúan con el jugador. Hereda de Personaje e incluye una instancia de la clase Pregunta (ver más abajo).
- **Clase Borde**
    - Hereda de personaje y se utiliza para "dibujar" las paredes y muebles que aparecen en el mapa para poder controlar las colisiones. Las posiciones y tamaños de dichos objetos se cargan desde el fichero "objetos.dat" que contiene posX, posY, ancho y alto de cada instancia por cada línea.

- **Clase Pregunta**
    - Aquí definimos los diferente puzles a resolver. En este momento el constructor sólo permite del tipo "pregunta/respuesta a elegir". Pero sería posible, con unos mínimos cambios incluir puzles más complejos a las preguntas.
    - El proceso es el siguiente:
        - El jugador colisiona un NPC.
        - El NPC comprueba si el jugador tiene en su inventario el objeto que es requisito para el puzle. En caso negativo salta un aviso.
        - En caso de que el NPC no tenga nada en su inventario se lanza un aviso.
        - El NPC plantea el puzle.
        - Si el jugador lo solventa el NPC "entrega" el objeto tiene en su inventario al jugador.

CONSIDERACIONES GENERALES A LA HORA DE MODIFICAR EL CÓDIGO.

- El objeto encargado de dibujar los objetos es la instancia correspondiente de la clase Hardware. Por comodidad (y por algo de lógica) a los objetos tipo Personaje también se les otorga (a través del método correspondiente) la capacidad de dibujarse a sí mismos. Pero en este caso es necesario pasar el objeto Hardware por parámetro.

- Las imágenes se superponen en el orden en que se dibujan, como en HTML. La primera es la capa (o layer) más abajo.

- Para manejar los desplazamientos y las colisiones hay tres métodos en la clase Juego.
    - *AjustarPosicion(short paso)*: Que ajusta las posiciones de todos los objetos de acuerdo al incremento (o decremento) pasado en paso.
    - *ComprobarTecla()*: Que lanza AjustarPosición() en su uso normal. También maneja la animación del jugador y prepara el menú para ser visualizado.
    - *ComprobarColisiones()*: Que verifica si el jugador se solapa con alguna imagen y lanza AjustarPosición() moviendo en jugador un paso hacia atrás si es necesario.

- Como he comentado al principio del manual. En el fuente se hayan intercalados los comentarios necesarios que explican cada clase, los atributos y las zonas más delicadas del código.