Primera Presentación Intermedia.Diseño de Videojuegos

Universidad de Cádiz

28 de abril de 2009

Índice I

Índice II

Cambios Planificación

- Se ha actualizado en trabajo realizado por cada componente.
- Francisco: Ingeniería aplicada (Casos de Uso). Planificación. Cambios metodología de juego.
- Álvaro: Clases. Sprites jugador y enemigos. GameLoop.
- Leandro: Presentación. Tiles. Primer código. Ingeniería aplicada.

Gantt

Gráfica Gantt

WBS	Nombre	Obra	Semana 11, 2009	Semana 12, 2009	Semana 13, 2009	Semana 14, 2009	Semana 15, 2009	Semana 16, 2009	Semana 17, 2009	Semana 18, 2009	Semana 19, 2009	Semana 20, 2009	Semana 21, 2009	Semana 22, 2009
			4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31											
1	Elección Videojuego	6d	Fran, Álvaro, Leandro											
2	**Planteamiento**	**3d**												
2.1	Historia	1d	Leandro											
2.2	Funcionalidad	1d												
2.3	Objetos	1d												
3	**Diagrama Gantt**	**3d**												
3.1	Identificación Tareas	2d												
3.2	Asignación Tareas	1d												
4	C.U. Menú Ppal. y Pausa	2d	Leandro											
5	Búsqueda Fondos	5d	Leandro											
6	Diseño Objetos	5d	Fran											
7	**Presentación 1**	**6d**												
7.1	P1.Recopilación	1d												
7.2	P1.Redacción	3d												
7.3	P1.Revisión	2d			Fran, Álvaro, Leandro									
8	Exposición Proyecto	1d			Fran, Leandro									
9	Diseño Personaje	3d			Álvaro									
10	Diseño Partes Escenario	3d			Fran									
11	**Programación**	**48d**												
11.1	Implementación Clases	3d				Álvaro								
11.2	Algoritmo Principal	3d												
11.3	SDL	5d					Leandro							
11.4	Programación	37d												
12	**Cambios Proyecto**	**4d**												
12.1	Revisión Planteamiento	2d						Fran						
12.2	Nuevas Ideas	2d						Fran, Álvaro						
13	C.U. Jugar Partida	2d							Fran, Leandro					
14	**Presentación 2**	**4d**												
14.1	P2.Recopilación	1d								Fran, Álvaro, Leandro				
14.2	P2.Redacción	2d								Leandro				
14.3	P2.Revisión	1d								Fran, Álvaro, Leandro				
15	1ª Presentación Intermedia									Álvaro, Leandro				
16	2ª Presentación Intermedia													
17	**Instrucciones**	**4d**												
17.1	Inst.Redacción	2d												
17.2	Inst.Diseño	2d												
18	Pruebas	4d												
19	Entrega Final													

Figura: Gantt

Gantt 2

WBS	Nombre	Inicio	Fin	Obra	Prioridad	Terminado	Coste
1	Elección Videojuego	Mar 4	Mar 11	6d		100%	
2	**Planteamiento**	Mar 12	Mar 16	3d			
2.1	Historia	Mar 12	Mar 12	1d		100%	
2.2	Funcionalidad	Mar 13	Mar 13	1d		100%	
2.3	Objetos	Mar 16	Mar 16	1d		100%	
3	**Diagrama Gantt**	Mar 12	Mar 16	3d			
3.1	Identificación Tareas	Mar 12	Mar 13	2d		100%	
3.2	Asignación Tareas	Mar 16	Mar 16	1d		100%	
4	C.U. Menú Ppal. y Pausa	Mar 12	Mar 13	2d		100%	
5	Búsqueda Fondos	Mar 12	Mar 18	5d		100%	
6	Diseño Objetos	Mar 12	Mar 18	5d		100%	
7	**Presentación 1**	Mar 19	Mar 25	6d			
7.1	P1.Recopilación	Mar 19	Mar 19	1d		100%	
7.2	P1.Redacción	Mar 20	Mar 24	3d		100%	
7.3	P1.Revisión	Mar 25	Mar 25	2d		100%	
8	Exposición Proyecto	Mar 25	Mar 25	1d		100%	
9	Diseño Personaje	Mar 25	Mar 30	3d		100%	
10	Diseño Partes Escenario	Mar 25	Mar 30	3d		100%	
11	**Programación**	Mar 30	Jun 1	48d			
11.1	Implementación Clases	Mar 30	Apr 2	3d		30%	
11.2	Algoritmo Principal	Mar 30	Apr 2	3d		10%	
11.3	SDL	Apr 2	Apr 9	5d		20%	
11.4	Programación	Apr 9	Jun 1	37d		10%	
12	**Cambios Proyecto**	Apr 15	Apr 17	4d			
12.1	Revisión Planteamiento	Apr 15	Apr 16	2d		100%	
12.2	Nuevas Ideas	Apr 17	Apr 17	2d		100%	
13	C.U. Jugar Partida	Apr 20	Apr 20	2d		100%	
14	**Presentación 2**	Apr 23	Apr 27	4d			
14.1	P2.Recopilación	Apr 23	Apr 23	1d		100%	
14.2	P2.Redacción	Apr 23	Apr 27	2d		100%	
14.3	P2.Revisión	Apr 27	Apr 27	1d		100%	
15	1º Presentación Intermedia	Apr 29	Apr 29				
16	2º Presentación Intermedia	May 27	May 27				
17	**Instrucciones**	Jun 1	Jun 2	4d			
17.1	Inst.Redacción	Jun 1	Jun 2	2d		0%	
17.2	Inst.Diseño	Jun 2	Jun 2	2d		0%	
18	Pruebas	Jun 1	Jun 2	4d		0%	
19	Entrega Final	Jun 3	Jun 3				

Recursos

Nombre	Nombre corto	Tipo	Grupo	Correo electrónico	Coste
Francisco	Fran	Obra	Grupo	fran_play@hotmail.com	0
Leandro	Leandro	Obra	Grupo	leandropastrana@gmail.com	0
Álvaro	Álvaro	Obra	Grupo	alvaro_barchilon@hotmail.com	0

Figura: Gantt

Revisión

El juego que estamos haciendo consta de dos partes claramente diferenciadas:

- Parte 1: Tipo Puzzle. El jugador debe resolver un determinado puzzle contrarreloj para salir del recinto a la vez que encuentra un objeto necesario para escapar.
- Parte 2: Tipo Estrategia. El jugador debe alternar el turno con los enemigos controlados por el juego para dirigirse, sin ser atrapado, hasta el "Punto de Escape", donde utilizará el objeto antes recogido para escapar.

Revisión

Comprobamos como en la primera parte del juego se le plantea al jugador el reto de resolver un puzzle en un tiempo determinado, por lo que la huída con éxito o no del recinto reside en la pericia del jugador. En cambio, en la segunda parte, la consecución del objetivo (llegar al "Punto de Escape" sin ser atrapado) radica más en la suerte que en la experiencia del jugador, ya que sólo se tienen en cuenta los números obtenidos en las tiradas de los dados, que marcan el número de movimientos disponibles por turno. Surge, por tanto, la necesidad de añadir nuevos elementos al juego que permitan dotar al jugador de más opciones para conseguir escapar en esta segunda fase y que no sólo se tenga en cuenta el factor suerte.

Cambios

- Cambio 1: Se añaden campos de visión a los enemigos de forma tal que éstos no sepan nuestra posición exacta a no ser que estemos dentro de él. De esta forma, los enemigos se dirigiran hasta nosotros para intentar atraparnos cuando estemos dentro de su campo de visión o se moverán de manera aleatoria por el escenario cuando no sepan dónde nos encontramos.
  Los campos de visión consisten en unas celdas que tengan una distancia igual o menor a la elegida por los programadores hasta cada uno de los enemigos y se representarán con un color distinto de las celdas que no pertenezcan a ningún campo de visión.

Cambios

- Cambio 2: Se introducen elementos en el escenario tales como árboles, piedras, ... que permitan al jugador esconderse de los enemigos de forma que si uno de éstos elementos se interpone entre el jugador y el enemigo el campo de visión se vea reducido.

Cambios

Podemos comprobar como gracias a estos cambios se añaden nuevos toques de estrategia en esta segunda parte del juego, ya que el jugador podría probar suerte y dirigirse directamente al "Punto de Escape" o intentar llegar a él escondiéndose de los campos de visión enemigos. Además, gracias a los campos de visión, se facilita la tarea de cambiar entre niveles de dificultad agrandando o disminuyendo el campo de visión de los enemigos.

Interfaces Externa

- Se utilizarán las librerias SDL bajo c++, que proporciona la interfaz entre el hardware y la aplicación.
- Se utilizará una resolución 800x600, pudiendo cambiarse a pantalla completa.
- El interfaz hardware de entrada para el manejo será el teclado.

Interfaces Externa

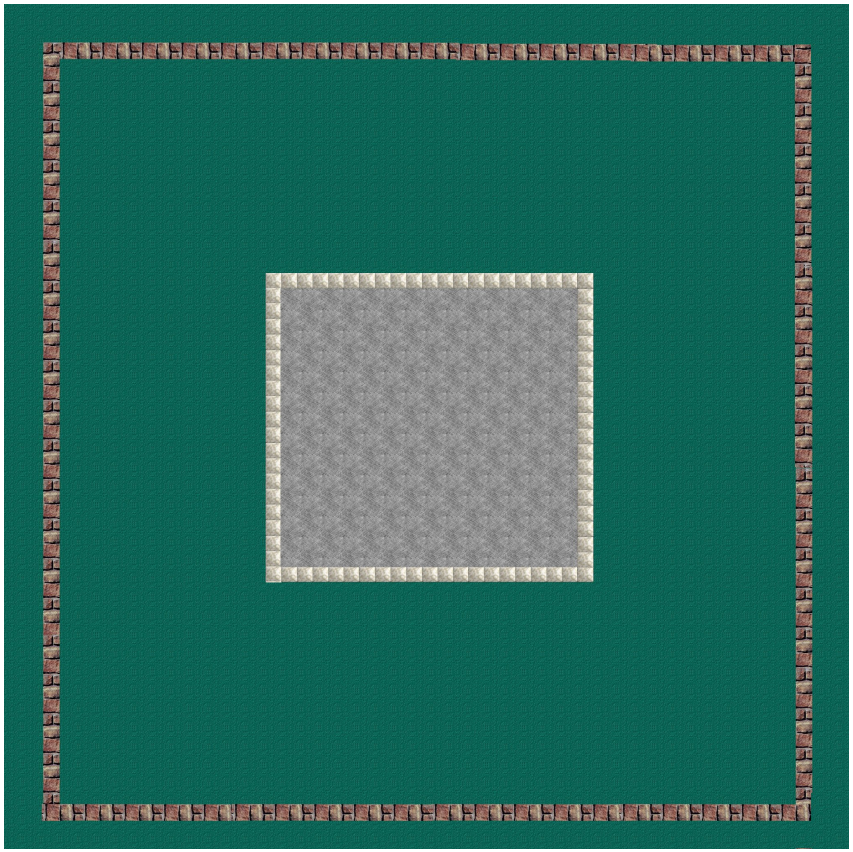
Encontramos 3 ventanas de acción con el ususario. Menú principal:

Interfaces Externa

Menú de carga de partida salvada:

Interfaces Externa

Ventana de juego:



Pseudocódigo

```
Inicio
 Mostrar_Menu();
 Mientras(true){
 Switch(opcion){
    case nueva_partida:
       IniciarPartida();
       JugarPartida();
       FinalizarPartida();
    case cargar_partida:
       MostrarGuardadas();
       si (EligeUna()){
          CargarPartida();
         JugarPartida();
         FinalizarPartida();
        }
       si_no{
        }
    case salir:
       SalirPrograma();
     }
}
Fin
– JugarPartida() –
Inicio
 Mientras(true){
    GestionarGraficos();
    GestionarEntrada();
    Procesar();
  }
Fin
```

Pseudocódigo

- GestionarGraficos() es una funcion que consiste en pintar el Laberinto o las posiciones de los guardias y el prisionero en cada caso, en funcion a la tecla pulsada. Si estamos dentro del laberinto, hay que pintar el tiempo, el objeto obtenido, y la salida abierta cuando se resuelve el puzzle. Si estamos fuera, hay que pintar la posicion de los guardas, su angulo de vision, los objetos que obstaculizan el angulo, y nuestra posicion, ademas de los dados.
- GestionarEntrada() se encarga de procesar las teclas que ha pulsado el jugador, si pulsa la tecla de menu, si pulsa direccion, si pulsa la tecla de tirar dados, etc... y pasar esos datos a la funcion Procesar().

Pseudocódigo

- Procesar() se encarga del videojuego en si. Actua en consecuencia, segun donde se encuentra. Si estamos dentro, por mucho que se pulse la tecla de tirar dados, no se obtendra tirada. Si se pulsa menu, hay que mostrar el menu. Si se pulsa una tecla de direccion, y hay un obstaculo que lo impide (pared, camion, roca...) no se puede avanzar. Ademas hay que tener en cuenta que si se acaba el tiempo, la partida habra concluido.
- FinalizarPartida() lo que hace es liberar recursos y mostrar las puntuaciones obtenidas.

GameLoop

Menú de carga de partida salvada:

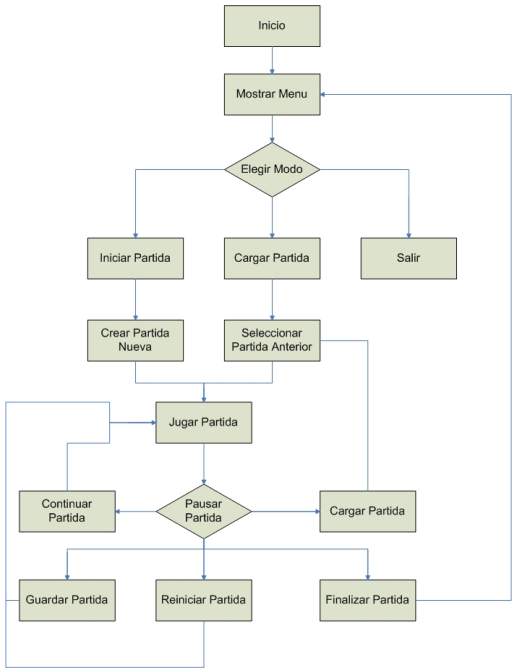


Figura: Menú de carga

GameLoop

Menú de carga de partida salvada:

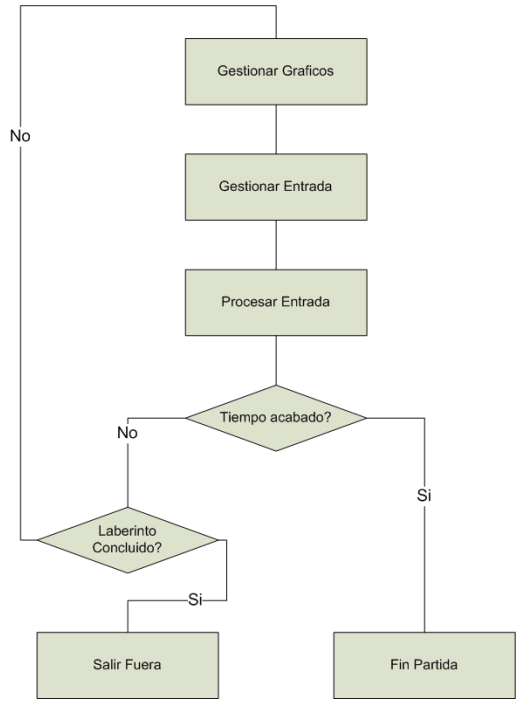


Figura: Menú de carga

GameLoop

Menú de carga de partida salvada:

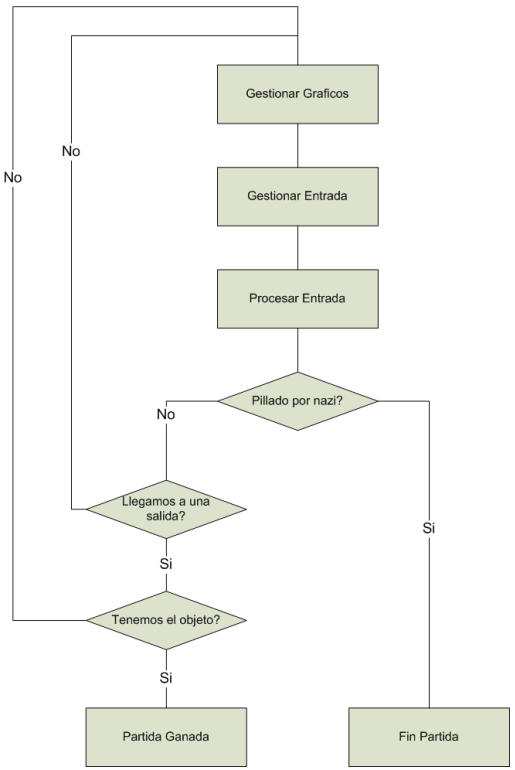


Figura: Menú de carga

Requisitos Funcionales

El juego consta de los siguientes requisitos principales

- Interacción del usuario con el juego mediante el teclado.
- Selección de nueva partida en la pantalla de menú.
- Salvar la partida actual en cualquier momento del juego.
- Cargar una partida anterior en la pantalla de menú.
- Salir de la aplicación en cualquier momento.

Requisitos de rendimiento

Al tratarse de un juego en 2D solo tendremos en la memoria cada uno de los píxeles que forman la imagen, a lo que se añaden gráficos sencillos, así que el juego necesitará pocos requisistos de memoria RAM y memoria de video. Estos requisitos se irán refinando a medida que la aplicación se desarrolle.
Respecto a la portabilidad en principio el código actual funciona bajo Linux y Winxx.

Desarrollo del juego

El Sistema carga el primer nivel de juego o el selccionado por el usuario al cargar partida. El usuario resuelve los puzzles y realiza las acciones necesarias para pasar de nivel, estas acciones son:

Desarrollo del juego

1. El Usuario mueve al personaje para empezar la partida y la cuenta atrás.
2. El Usuario resuelve el puzzle del interior del recinto antes de finalizar la cuenta atrás.
3. El Usuario entonces puede recoge el objeto necesario para escapar antes de finalizar la cuenta atrás.
4. El Usuario sale al patio del recinto antes de finalizar la cuenta atrás.
5. El Usuario "tira los dados" y avanza hasta el "Punto de Escape" tantas casillas como marquen éstos.
6. El Sistema "tira los dados" y mueve en una dirección aleatoria a los Guardas del patio tantas casillas como indiquen éstos, en busca del jugador. Si el usuario está en el campo de visión del jugador moverá hacia él.
7. El Usuario utiliza el objeto, escapa y consigue acabar el nivel.

Modelo basado en componentes

Se realizará un modelo basado en componentes. Este modelo sigue un modelo en espiral y evolutivo. Donde en cada iteración se van construyendo y refinando versiones del software más completas. Este modelo utiliza además componentes software ya existentes como es el caso de las librerías SDL. Se irán siguiendo las siguientes iteaciones:

Modelo basado en componentes

Se realizará un modelo basado en componentes. Este modelo sigue un modelo en espiral y evolutivo. Donde en cada iteración se van construyendo y refinando versiones del software más completas. Este modelo utiliza además componentes software ya existentes como es el caso de las librerías SDL. Se irán siguiendo las siguientes iteaciones:

- Planificación.

Modelo basado en componentes

Se realizará un modelo basado en componentes. Este modelo sigue un modelo en espiral y evolutivo. Donde en cada iteración se van construyendo y refinando versiones del software más completas. Este modelo utiliza además componentes software ya existentes como es el caso de las librerías SDL. Se irán siguiendo las siguientes iteaciones:

- Planificación.
- Análisis.

Modelo basado en componentes

Se realizará un modelo basado en componentes. Este modelo sigue un modelo en espiral y evolutivo. Donde en cada iteración se van construyendo y refinando versiones del software más completas. Este modelo utiliza además componentes software ya existentes como es el caso de las librerías SDL. Se irán siguiendo las siguientes iteaciones:

- Planificación.
- Análisis.
- Ingeniería.

Modelo basado en componentes

Se realizará un modelo basado en componentes. Este modelo sigue un modelo en espiral y evolutivo. Donde en cada iteración se van construyendo y refinando versiones del software más completas. Este modelo utiliza además componentes software ya existentes como es el caso de las librerías SDL. Se irán siguiendo las siguientes iteaciones:

- Planificación.
- Análisis.
- Ingeniería.
- Pruebas de software.

Modelo basado en componentes

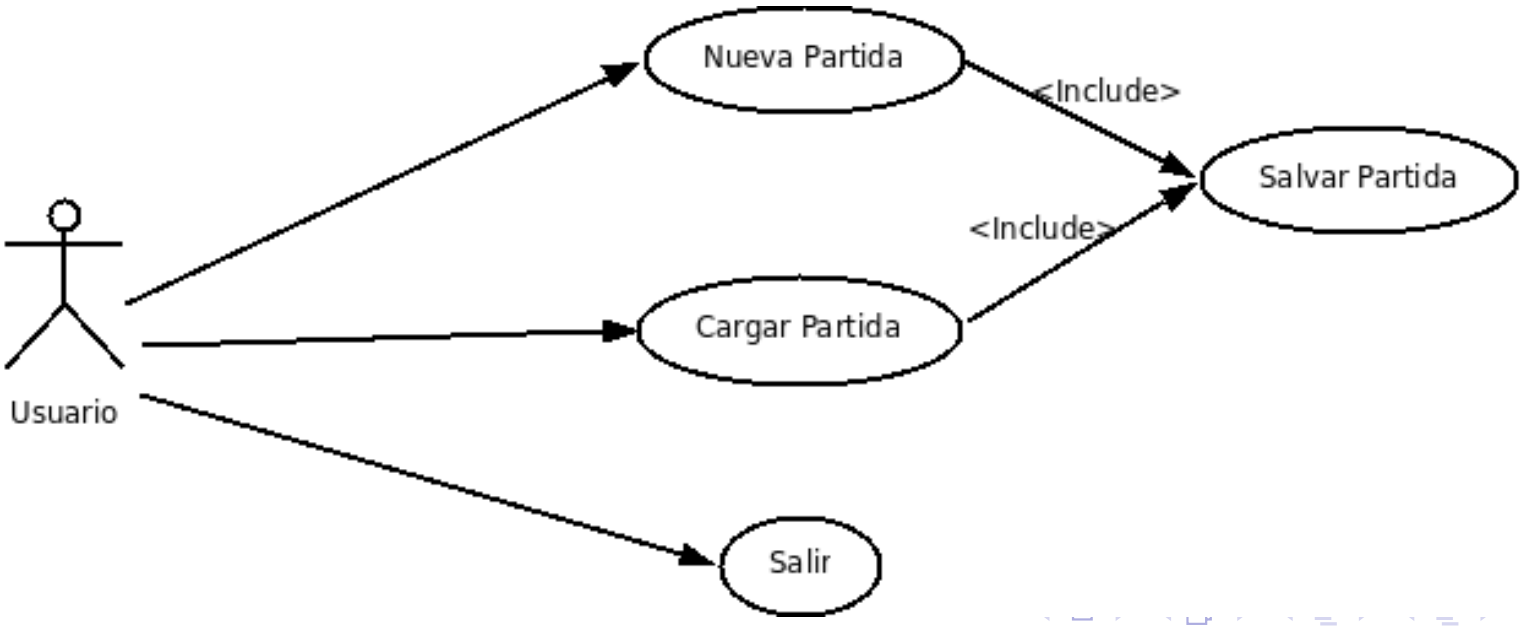
Se realizará un modelo basado en componentes. Este modelo sigue un modelo en espiral y evolutivo. Donde en cada iteración se van construyendo y refinando versiones del software más completas. Este modelo utiliza además componentes software ya existentes como es el caso de las librerías SDL. Se irán siguiendo las siguientes iteaciones:

- Planificación.
- Análisis.
- Ingeniería.
- Pruebas de software.
- Evaluación del producto.

Casos de Uso

Se identifican tres formas de interactuar el usuario con el sistema:

- **Nueva Partida:** Donde el usuario comienza una partida desde cero.
- **Cargar Partida:** Donde el usuario decide cargar una partida existente.
- **Salir:** Donde el usuario decide salir del juego.



Descripción CU Nueva Partida

DESCRIPCIÓN CASO DE USO: Comenzar Partida

- **Caso de uso:**Comenzar partida
- **Descripción:**Comienza una partida desde el primer nivel
- Actores: Usuario
- **Precondiciones:**Ninguna
- **Postcondiciones:** El usuario interactua con el videojuego
- **Escenario principal:**
  1. El usuario comienza partida
  2. El sistema carga el primer nivel
  3. El usuario interactua con el sistema

Descripción CU Nueva Partida

- **Escenario alternativo:**
  - ***a.**El usuario puede salir en cualquier momento
  - ***b.**El usuario puede salvar en cualquier momento:**INCLUDE: SALVAR PARTIDA**
  - ***c.**El usuario puede reiniciar en cualquier momento:**INCLUDE: REINICIAR**
  - **2a.**El sistema no puede cargar el primer nivel
    - El sistema termina la aplicación
  - **3a.**Usuario termina nivel
    - El sistema pasa al siguiente nivel
  - **3b.**Usuario termina nivel y es el último
    - El sistema muestra final de juego.

Descripción CU Cargar Partida

DESCRIPCIÓN CASO DE USO: Cargar Partida

- **Caso de uso:**Cargar partida
- **Descripción:**Cargar una partida salvada anteriormente
- Actores: Usuario
- **Precondiciones:**Existe una partida salvada
- **Postcondiciones:** Se carga la partida seleccionada
- **Escenario principal:**
  1. El usuario selecciona cargar una partida
  2. El sistema carga la partida seleccionada
  3. El usuario interactua con el sistema a partir de la partida cargada

Descripción CU Cargar Partida

- **Escenario alternativo:**
  - ***a.**El usuario puede salir en cualquier momento
  - ***b.**El usuario puede salvar en cualquier momento. **INCLUDE: SALVAR PARTIDA**
  - ***c.**El usuario puede reiniciar en cualquier momento:**INCLUDE: REINICIAR**
  - **2a.**El sistema no puede cargar la partida
    - El sistema termina la aplicación
  - **2b.**La partida no existe
    - El sistema indica error
  - **3a.**Usuario termina nivel
    - El sistema pasa al siguiente nivel
  - **3b.**Usuario termina nivel y es el último
    - El sistema muestra final de juego.

Descripción CU Salir

DESCRIPCIÓN CASO DE USO: Salir

- **Caso de uso:**Salir
- **Descripción:**La apilación se cierra
- Actores: Usuario
- **Precondiciones:**ninguna
- **Postcondiciones:** Se sale de la aplicación
- **Escenario principal:**
  1. El usuario selecciona salir de la aplicación
  2. El sistema cierra la aplicación

Descripción CU Salvar Partida

DESCRIPCIÓN CASO DE USO: Salvar Partida

- **Caso de uso:**Salvar partida
- **Nivel:**Subfunción
- **Descripción:**Se guarda la partida actual
- Actores: Usuario
- **Precondiciones:**Existe una partida en ejecución
- **Postcondiciones:** Se guarda la partida actual desde el comienzo
- **Escenario principal:**
  1. El usuario selecciona salvar partida
  2. El sistema salva carga la partida seleccionada

Descripción CU Salvar Partida

- **Escenario alternativo:**
  - ***a.**El usuario puede salir en cualquier momento
  - **2a.**El sistema no puede salvar la partida
    - El sistema continua la partida actual

Descripción CU Reiniciar

DESCRIPCIÓN CASO DE USO: Reiniciar

- **Caso de uso:**Reiniciar
- **Nivel:**Subfunción
- **Descripción:**Se reinicia el nivel actual
- Actores: Usuario
- **Precondiciones:**Existe una partida en ejecución
- **Postcondiciones:** Se reinicia el nivel actual
- **Escenario principal:**
  1. El usuario selecciona reiniciar nivel
  2. El sistema reinicia el nivel actual

C.O.: Nueva Partida

- Operación: Nueva Partida
- Responsabilidades: Carga una nueva partida desde el primer nivel de juego.
- Precondiciones: Ninguna.
- Postcondiciones: Carga en memoria el primer nivel de juego.
  Si no exite muestra un error y termina la aplicación.

C.O.: Cargar Partida

- Operación: Cargar Partida
- Responsabilidades: Carga la partida seleccionada por el usuario.
- Precondiciones: Debe de existir al menos una partida salvada.
- Postcondiciones: Carga el juego desde la partida selccionada. Si no existe el archivos muestra el error y la aplicación termina.

C.O.: Salir

- Operación: Salir
- Responsabilidades: Se termina la ejecución de la aplicación.
- Precondiciones: La aplicación está en ejecución.
- Postcondiciones: Se libera la memoria que ocupa la aplicación.

C.O.: Salvar

- Operación: Salvar.
- Responsabilidades: Se guarda el estado de la partida actual.
- Precondiciones: Existe una partida en ejecución.
- Postcondiciones: Se crea un archivo con el estado completo del juego.
  Si no peude crear el archivo la aplicación continua.

C.O.: Reiniciar

- Operación: Reiniciar
- Responsabilidades: Se vuelve al comienzo del nivel actual.
- Precondiciones: Exite una partida en ejecución.
- Postcondiciones: El nivel vuelve al estado principal.
  Si falla la aplicación se cierra.

DSS: Nueva Partida

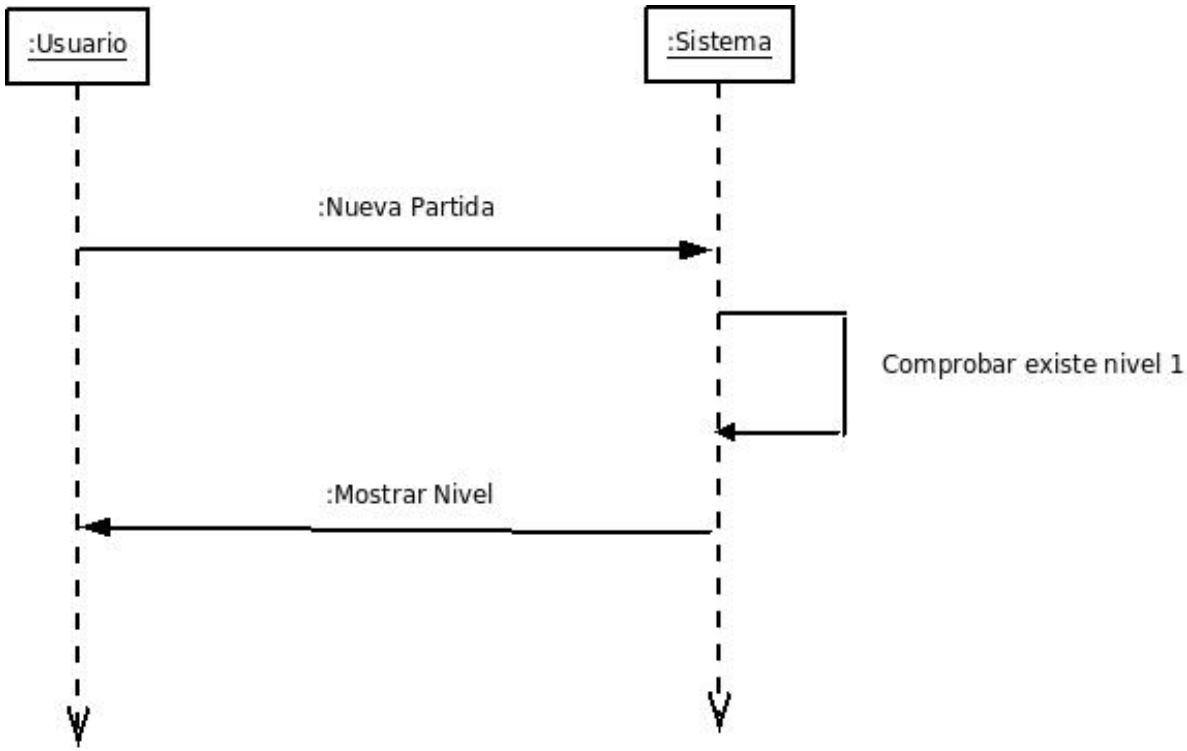


Figura: Nueva Partida

DSS: Cargar Partida

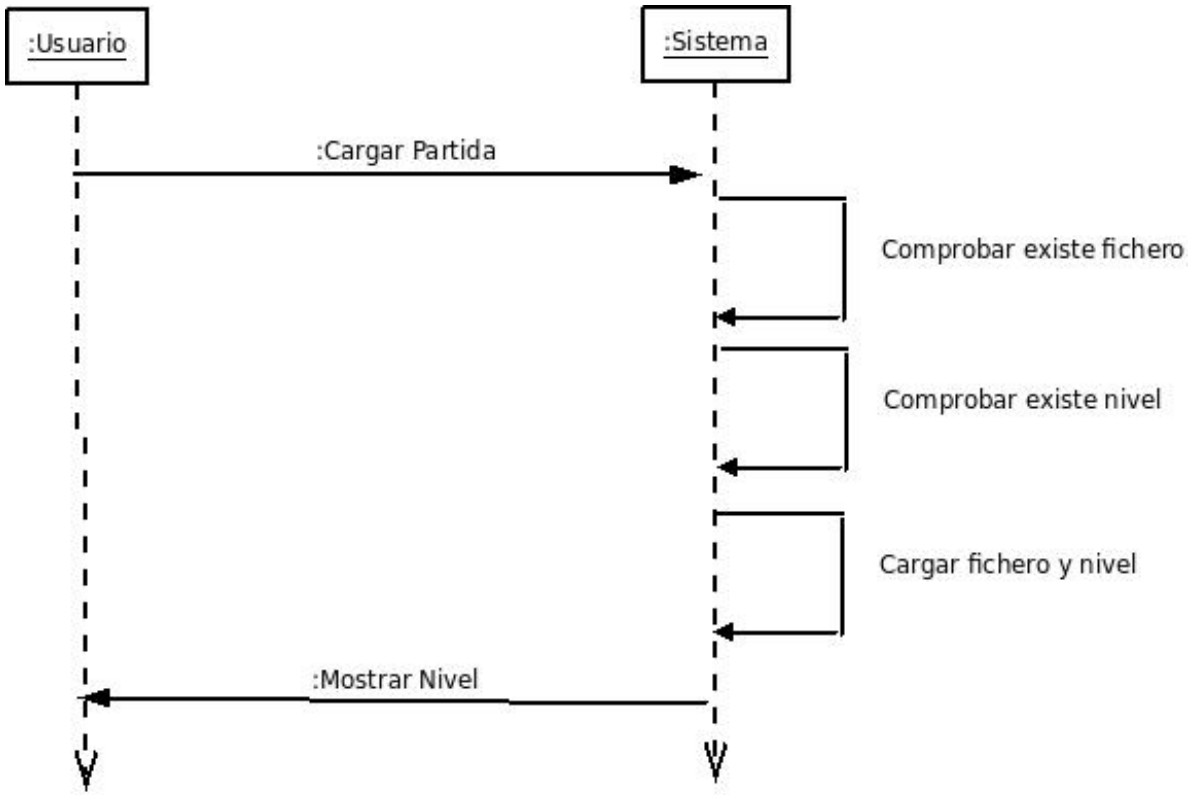


Figura: Cargar Partida

DSS: Salir

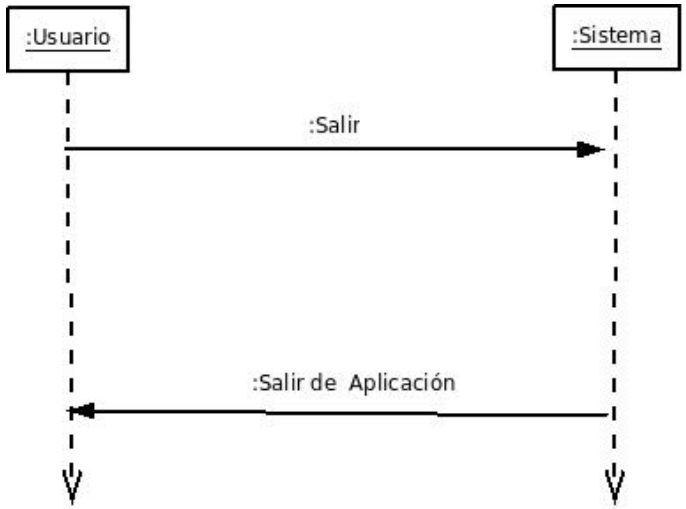


Figura: Salir

DSS: Salvar

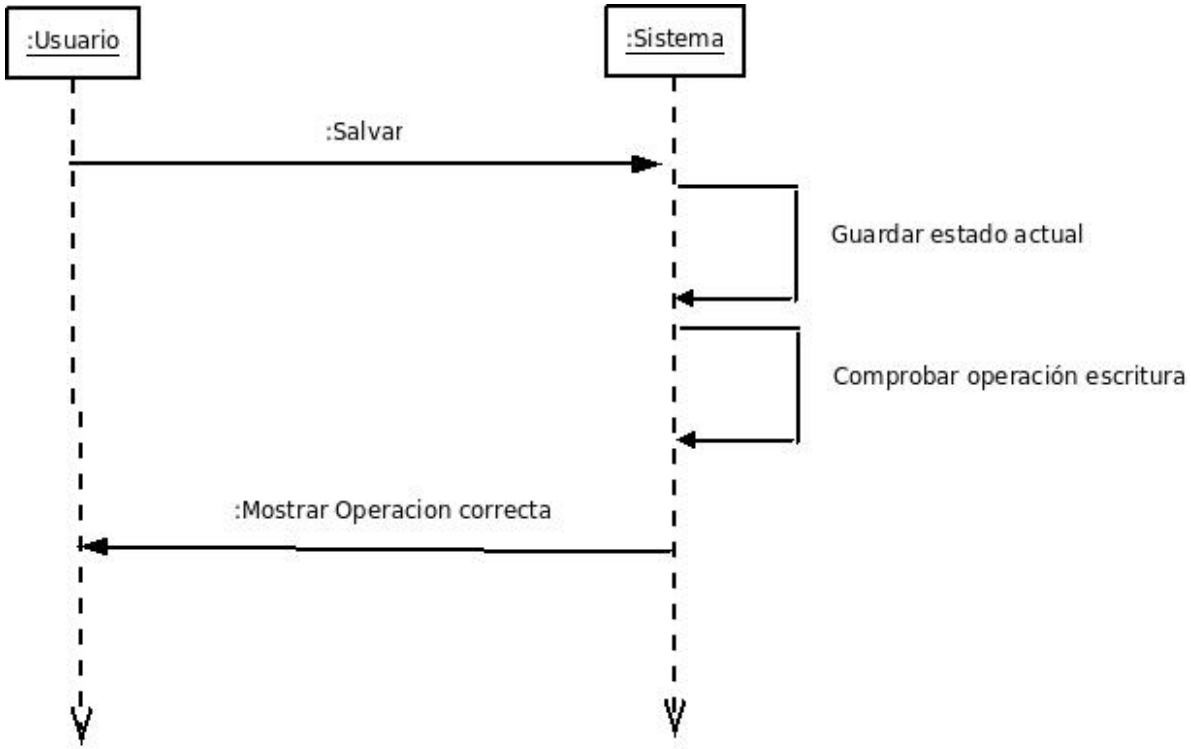


Figura: Salvar

DSS: Reiniciar

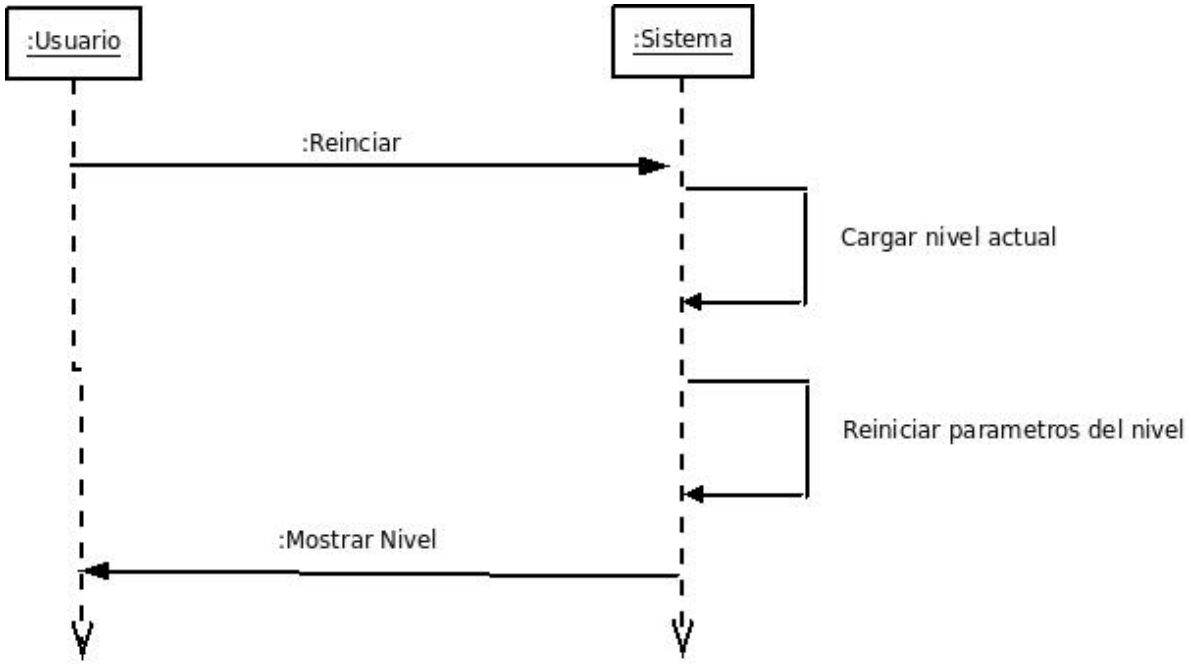


Figura: Reiniciar

Implementación

Se ha escogido un sistema de coordenadas representado por una matriz, en esta matriz se dibuja el fondo junto con el personaje y los objetos y enemigos. Sobre esta matriz se realiza un procesamiento mediante la cual se actualiza de acuerdo a las acciones realizadas.
EL propio personaje, cada objeto y enemigos mantienen su posición en la matriz de coordenadas para poder aplicar su posición cuando se actualize esta matriz.
Respecto a la implementación se ha escogido en primer lugar una resolución 800x600 y gráficos de 25 pixeles. Con ello obtenemos una matriz para cargar los tiles de 32x24 casillas, en cada una de las cuales se cargara un tile correspondiente formando de esta forma la imagen de fondo.

Implementación

Una vez cargada la imagen de fondo y cargado el personaje, objetos y enemigos aplicando transparencias (concretamente al valor 255 de fondo), se implementan las siguientes funciones:

- DibujarHeroe(xHeroe,yHeroe): La cual dibuja el personaje de acuerdo a las teclas y coordenadas.
- DibujarBloque(xBloque,yBloque): La cual dibuja el bloque de acuerdo a las teclas y coordenadas.
- MoverHeroeDentro(): La cual actualiza las coordenadas del personaje de acuerdo a las teclas y si fuese necesario actualiza las coordenadas del bloque. Además esta función limita la zona de acción sin que el personaje o el bloque salgan de la pantalla.

Revisión

El seguimiento del trabajo realizado debido a los horarios y disponibilidad geográfica de los participantes se ha llevado a cabo en un 80 % de forma virtual, es decir se han divido las tareas y cada uno ha aportado su trabajo. El resto de componentes se encarga de revisar el trabajo de cada compañero.

- Cada participante ha subido su trabajo a la forja o en su defecto en caso de archivos no definitivos se han enviado al e-mail.
- Las fechas de entrega se han cumplido en casi todos los casos.
- Queda pendiente una nueva revisión de la asignación y ejecución de tareas.