1. ENTREVISTA DIRIGIDO A USUARIOS DEL SISTEMA

**Proyecto:** Juego de Pacman

**Doc.:** EntrevistaXX **Autor:** Entrevistador

**Fecha:** ##/##/## **Entrevistado:** Entrevistado 1Usuario del sistema

1. Te gusta el juego de Pacman?

R. Si

1. Como empezó a jugar Pacman?

R. De pequeño vi el juego y lo fácil q era manejarlo.

1. Desde que plataforma utilizas el juego de Pacman?

R. Actualmente lo uso desde mi computadora.

1. Hasta que nivel llegaste?

R. Bueno, el máximo nivel que alcance fue el 5.

1. Que es lo que no te gusta del juego?

R. Lo que no me gusta es que solo se manejan un solo tipo de laberinto.

1. Que te gustaría implementar o cambiar del juego?

R. me gustaría implementar nuevos fantasmas y distintos laberintos.

1. Que atributo te gustaría implementar en el personaje de Pacman?

R. Me gustaría que Pacman salte sobre los rivales como en otras versiones.

1. Cuál fue tu puntuación mas alta en el video juego?

R. La verdad no me acuerdo con exactitud pero estuve entre los primeros en la tabla de marcadores.

1. Te gustaría una versión de Pacman exclusivo para android?

R. Me encantaría, lo estuve esperando desde hace mucho tiempo.

1. Desea usted poder compartir esta aplicación con personas fuera de su círculo social?  
   R. Si, me gustaría.
2. Te gustaría poder elegir el nivel del juego a tu gusto?

R. No

**Proyecto:** Juego de Pacman

**Doc:** EntrevistaXX **Autor:** Entrevistador

**Fecha:** ##/##/## **Entrevistado:** Entrevistado 2Usuario del sistema

1. Le gusta la música de ambiente del juego?

R. Sí.

1. Que opciones adicionales le gustaría implementar en el juego?

R. La opción de cambiar de nivel.

1. Que datos le pide el sistema al terminar el juego?

R. Solo me pide un nombre.

1. Tienes alguna sugerencia para mejorar el juego?

R. Por el momento no.

1. Con que frecuencia utilizas este juego?

R. De 2 a 3 veces durante el mes.

1. Alguna vez participo de algún torneo referente a este juego?

R. No me entere de ningún torneo referente a este juego.

1. Recomendarías el juego a un amigo o familiar?

R. Si.

1. Usted sabe de cuantos niveles esta conformado el sistema?

R. No.

1. Le gustaría implementar nuevas músicas al sistema?

R. Si, porque no.

1. Cuantas versiones de Pacman conoce?

R. solo dos, la versión original y la versión de Play Station 1.

1. Sabes quién invento el juego?

R. No.

b) **DOCUMENTACIÓN DE PROYECTOS SIMILARES.**

**PROYECTO**

**LABERINTO PACMAN**

**INTRODUCCIÓN:**

El siguiente proyecto consiste en la implementación en Java del típico juego conocido como Pacman. En el cual solo se basara en la implementación del laberinto y del jugador, pudiendo moverlo con las teclas de desplazamiento.

**OBJETIVO GENERAL:**

El objetivo del proyecto es desarrollar el juego de Pacman, usando objetos y realizando los movimientos en dos dimensiones. El juego consistirá, en una figura de Pacman que recorre un laberinto utilizando las teclas de desplazamiento para su movimiento.

**OBJETIVOS ESPECÍFICOS:**

- Utilización de Eclipse como IDE.  
- Utilización de las diferentes librerías de AWT y JPanel.   
- Modelar el movimiento de los objetos.  
- Permitir que el usuario controle a Pacman  
- Diseñar un algoritmo para el movimiento de pacman  
- Implementar los conceptos de sobrecarga de métodos, Polimorfismo y manejo de excepciones.

**PLANTEAMIENTO DEL PROBLEMA:**

El juego ya conocido como Pacman consta de un laberinto un jugador y unos fantasmas, en este caso solo nos basaremos en realizar el laberinto de Pacman y el jugador. El cual tiene por objetivo desplazarse por los pasillos del laberinto utilizando las teclas de desplazamiento. Una vez oprimida cualquier tecla el jugador se moverá de acuerdo a la dirección que se indique en las líneas del laberinto ya sea arriba, abajo, derecha o izquierda.

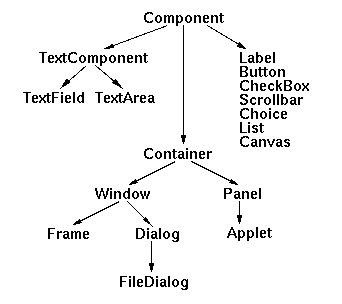
**LIBRERIAS:**

AWT

Permite hacer interfaces gráficas mediante artefactos de interacción con el usuario, como botones, menús, texto, botones para selección, barras de deslizamiento, ventanas de diálogo, selectores de archivos, etc. Y por supuesto despliegue gráfico general.

Jerarquía de clases de AWT

Cada una de las componentes de una ventana en AWT se representa mediante uno o más objetos de la aplicación. Estos objetos pertenecen a las clases que se observan en la siguiente jerarquía de clases para AWT:



SCANNER DEL PAQUETE JAVA.UTIL.

Un objeto Scanner permite a una aplicación leer datos ( números, strings ) introducidos por el usuario desde el teclado del ordenador, además de ofrecer muchas más utilidades que veremos en próximos artículos. Dichos datos los podemos guardar en variables.

IO.FILE

La clase File se usa para obtener información sobre archivos y directorios.   
Además la clase File permite crear y eliminar archivos y directorios.

Un objeto de la clase Java File representa un archivo o directorio.

**CLASES**

Se puede observar que el proyecto consta de las siguientes clases:

♦ Un Laberinto de juego: el cual contiene todos los elementos para jugar.

♦ Pacman: que puede caminar sobre el Laberinto

Laberinto: Contiene a todos los elementos que intervienen en el juego. Los elementos son los siguientes: Pacman

Borde

Pacman: Este objeto se va a estar moviendo dentro del laberinto

Borde: Son objetos los cuales van a detener al Pacman cuando tope en ellos y cambiará la dirección tanto de los objetos Fantasma y Fruta.

**CÓDIGO**

package game.packman;

import java.awt.Color;

import java.awt.Graphics2D;

import java.awt.event.KeyEvent;

import java.awt.image.BufferedImage;

import java.io.File;

import java.io.FileNotFoundException;

import java.io.IOException;

import java.util.ArrayList;

import java.util.Scanner;

import javax.imageio.ImageIO;

import org.game.engine.Game;

public class PackMan extends Game {

public static void main(String[] args) {

GameApplication.start(new PackMan ());

}

final int AVANCE = 2;

BufferedImage packman ;

int frame;

int reqDir, actDir;

int columna, fila;

int filas, columnas;

ArrayList<String> lineas = new ArrayList <String>();

public PackMan (){

//leer las lineas

try {

Scanner s = new Scanner(new File("maze.txt") );

int f = 0;

while (s.hasNextLine()){

String linea = s.nextLine();

lineas.add(linea);

if (linea.contains("5")){

fila = f;

columna = linea.indexOf('5');

}

f++;

}

s.close();

filas = lineas.size();

columnas = lineas.get(0).length();

width = columnas\*AVANCE;

height = filas\*AVANCE;

} catch (FileNotFoundException e1) {

// TODO Auto-generated catch block

e1.printStackTrace();

}

title = "PackMan";

frame = 0;

actDir = reqDir = KeyEvent.VK\_RIGHT;

// ancho = altura = 400

try {

packman = ImageIO.read(new File("images/packman.png") );

} catch (IOException e) {

e.printStackTrace();

}

}

@Override

public void keyPressed(KeyEvent e){

int key = e.getKeyCode();

if(37 <= key && key <= 40){

reqDir = key;

}

}

@Override

public void update() {

frame ++;

if (frame > 5) {

frame = 0;

}

if (mover(reqDir)== SUCCESS){

actDir = reqDir;

}else {

mover (actDir);

}

}

static int SUCCESS = 1, FAIL = 0;

private int mover (int recDir) {

//pocicion actual de pacman es (fila, columna)

switch (reqDir) {

case KeyEvent.VK\_LEFT: //37

if (columna > 0 && charAt(fila, columna-1) != '0'){

columna -= 1;

return SUCCESS;

}

break;

case KeyEvent.VK\_UP: //38

if (fila > 0 && charAt(fila-1, columna) != '0'){

fila -= 1;

return SUCCESS;

}

break;

case KeyEvent.VK\_RIGHT: //39

if (columna < columnas-1 && charAt(fila, columna+1) != '0'){

columna += 1;

return SUCCESS;

}

break;

case KeyEvent.VK\_DOWN: //40

if (fila < filas-1 && charAt(fila+1, columna) != '0'){

fila += 1;

return SUCCESS;

}

break;

}

return FAIL;

}

private char charAt(int fila, int columna) {

return lineas.get(fila).charAt(columna);

}

@Override

public void draw(Graphics2D g) {

g.setColor (Color.blue);

for (int f=0; f<filas; f++){

for (int c=0; c<columnas; c++){

if(charAt(f,c) != '0'){

g.fillRect (c\*AVANCE-14, f\*AVANCE-14, 28, 28);

}

}

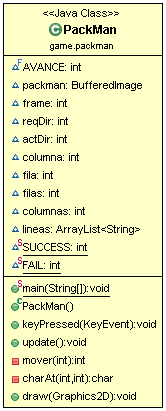
}

g.drawImage(packman.getSubimage((frame/2)\*30, (actDir -37)\*30, 28, 28), columna\*AVANCE-14 , fila\*AVANCE-14, null);

}

}

Diagrama de clases.



**CONCLUSIÓN:**

Durante la realización de este proyecto se han encontrado con diversos y variados problemas, lo cuales enumeramos a continuación:

En primer lugar, el enfrentamiento al lenguaje de programación como es Java ha sido una tarea ardua de tiempo y dedicación. Se han dedicado muchas horas de búsqueda de información en internet y de estudio de tutoriales para llegar a un mínimo manejo del lenguaje

La clase Pacman ha ocasionado mucha problemática por el hecho de que los objetos de dicha clase poseen un movimiento de acuerdo a los pasillos del laberinto arriba, abajo, derecha e izquierda y que no salga de los pasillos.

Dichos problemas se fueron resolviendo con la investigación que se realizó durante la realización del proyecto.

c) **REQUISITOS FUNCIONALES**

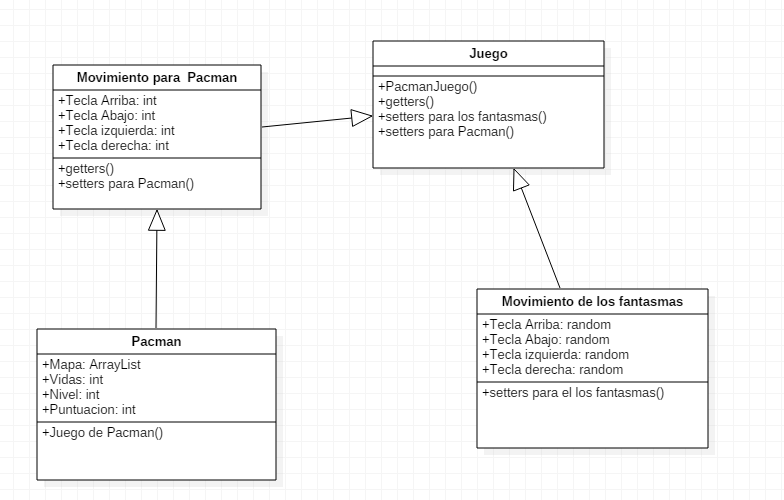
1. El sistema del juego deberá mostrar cuántas vidas le quedan al jugador.
2. El sistema del juego deberá mostrar los puntajes más altos de los jugadores.
3. El sistema del juego deberá permitir iniciar una nueva partida siempre y cuando el usuario lo quiera.
4. El sistema del juego deber volver a rellenar el laberinto después de cada nivel pasado.
5. El sistema del juego deberá de permitir iniciar una nueva partida.
6. El sistema del juego deberá de eliminar una vida cada vez que el personaje de Pacman choca con un fantasma.
7. El sistema del juego deberá permitir al usuario saber las puntuaciones más altas.
8. El sistema del juego deberá guardar los datos de los jugadores con mayor puntuación.
9. El sistema del juego deberá de permitir controlar el personaje de Pacman usando las teclas de flecha del teclado.
10. El sistema del juego deberá proporcionar frutas extras en el laberinto para así poder elevar más la puntuación del jugador.
11. El sistema del juego deberá reiniciar la puntuación cuando se inicia una nueva partida.
12. El sistema del juego deberá de reproducir la música clásica de Pacman mientras este se ejecuta.
13. El sistema del juego ofrecerá el juego de 1 a 2 players.
14. El sistema del juego dispondrá de varios niveles que aumentara la dificultad cada vez se supere cada nivel.
15. El sistema del juego contara con dos personajes, los fantasmas y Pacman.
16. 16.El sistema deberá de funcionar de manera impecable tanto en el descuento de vidas como en el registro de puntuaciones.
17. El sistema deberá contar con el mismo laberinto en todos los niveles.
18. El sistema deberá de contar con las mismas vidas cada vez que se empieza una nueva partida.

**REQUISITOS NO FUNCIONALES**

1. El sistema del juego deberá de ser multiplataforma.
2. El sistema del juego deberá de ser compatible con diferentes resoluciones de pantalla.
3. El sistema del juego deberá de trabajar en un rendimiento estable.
4. El sistema del juego deberá de otorgar la seguridad correspondiente al momento de registrar datos.
5. El sistema del juego deberá de ser fácil de entender e intuitivo.
6. El sistema del juego deberá de gestionar la base de datos.
7. El sistema del juego deberá de interactuar con el usuario mediante un menú interfaz.
8. El sistema del juego deberá de contar con requisitos mínimos para su funcionamiento en la pc.
9. El sistema del juego deberá de ser portable para un mejor manejo.
10. El sistema del juego tendrá un tiempo estándar para iniciar y reiniciar el juego.
11. El sistema del juego contara con módulos de ayuda en línea.
12. El sistema del juego no deberá de tener errores al ejecutarse.
13. El sistema del juego no deberá de tardar en iniciar una nueva partida.

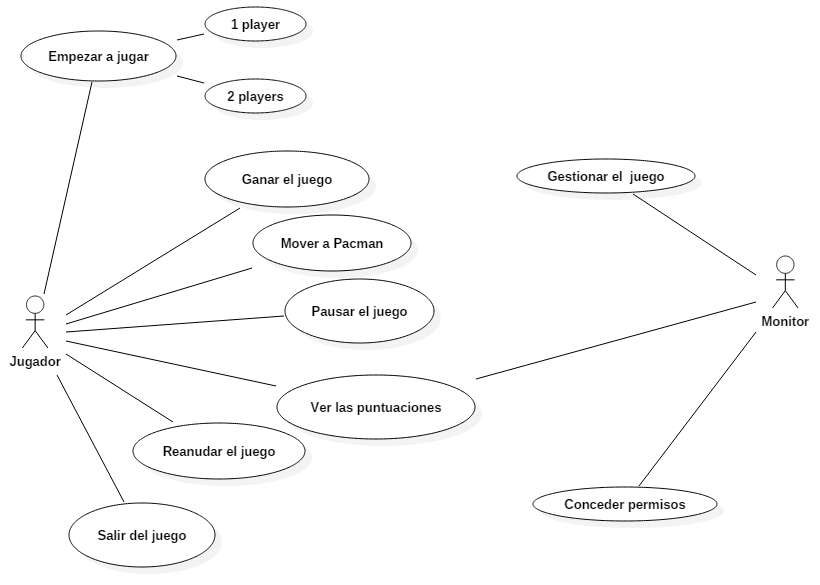
d) MODELO DEL DOMINIO (diagrama de clases inicial)

starUML



e) MODELO DEL NEGOCIO (diagrama de casos de uso)

starUML



f) ESPECIFICACIÓN DE CADA CASO DE USO (precondición, camino normal, caminos alternativos, post-condición)

|  |
| --- |
| CU: Iniciar juego |
| Actores: Jugador |
| Precondición: Entrar al juego |
| Camino Normal:   1. El jugador selecciona jugar. 2. El jugador selecciona partida de uno o dos Players. |
| Caminos alternos:   1. El jugador no selecciona ninguna opción. 2. El jugador selecciona mal la opción. |
| Post condición:  El jugador podrá jugar de uno a dos en turnos individuales si asi lo desea. |

|  |
| --- |
| CU: Gestionar el juego |
| Actores: Monitor |
| Precondición: ingresar al sistema |
| Camino normal:   1. El Monitor selecciona iniciar el sistema. 2. El Monitor le dará a conocer entre jugar de 1 a 2 players. |
| Caminos alternos:   1. El monitor mostrara las puntuaciones más altas. 2. El monitor actualizara base de datos. |
| Post condición:  El monitor reiniciara el sistema en caso de que el usuario no elija una opción en un tiempo determinado. |

g) REQUISITOS NO FUNCIONALES (requisitos adicionales)

* DOCUMENTACION

Toda la documentación acerca del juego estará almacenada en una base de datos accesible a cualquier usuario y en el idioma inglés-español.

* RECURSOS

Para un correcto funcionamiento del sistema se requiere que tenga las

Siguientes características, buena memoria RAM, y un disco duro lo

Suficiente para guardar los datos del usuario.

* JUGABILIDAD

El juego debe de contar con un menú amigable y fácil de entender de tal forma que el usuario pueda disfrutar mejor la aplicación.

* FUNCIONALIDAD

El sistema deberá de contar con un sistema modular para permitir posteriores actualizaciones del software.

* SOPORTE

El sistema deberá de poseer de un óptimo desempeño, tratándose de un juego que ordena y recibe órdenes de forma simultánea, puesto que un mínimo retraso afectaría de gran manera la jugabilidad.

* INTEGRIDAD

El sistema deberá tener una alta integridad y sobre todo fácil de instalar.

* INTERFAZ

La interfaz del sistema del juego debe de ser entorno Linux o Windows.

h) ESBOZOS DE INTERFACES DE USUARIO