TESIS PUCP

1. Anexo 1: Catálogo de historias de usuario

Bajo la metodología utilizada, y adaptada al proyecto, este catálogo presentará las funciones de la herramienta que permita implementar las reglas de un videojuego RPG/RTS de dos dimensiones. Adicionalmente sirven como unidad de trabajo para las estimaciones y la división del trabajo en incrementos.

A continuación se presentan las historias de usuario.

Código	Historia de usuario	Puntos	Sprint	Acumulado de puntos
1	Será posible crear un escenario de dos dimensiones mediante un archivo de configuración.	1	Sprint 1	22.5
2	Será posible configurar las dimensiones de un escenario mediante el archivo de configuración de escenario.	0.5		
3	Será posible configurar la distribución de las celdas del escenario mediante caracteres en el archivo de configuración de escenario.	2		
4	Será posible crear la distribución de los edificios en un escenario mediante el archivo de configuración de la distribución de edificios.	1		
5	Será posible configurar los accesos a los escenarios internos abriéndolos o cerrándolos.	0.5		
6	Será posible configurar la zona de entrada al escenario interno y la orientación(es) del personaje.	1		
7	Será posible configurar la posición inicial de un jugador al entrar a un escenario interno.	0.5		

Tesis publicada con autorización del autor
No olvide citar esta tesis

8	Será posible configurar el desplazamiento al entrar a un escenario interno, generando que el personaje camine automáticamente.	1		
9	Será posible crear un escenario interno perteneciente a un edificio mediante un archivo de configuración.	1		
10	Será posible configurar la distribución de elementos y si posición en un escenario interno, identificándolos con un carácter.	0.5		
11	Será posible crear un archivo con la configuración del escenario interno de dos dimensiones.	1		
12	Será posible configurar la zona de salida del escenario interno a un escenario externo la orientación(es) del personaje. .	1		
13	Será posible configurar la posición inicial de un jugador al salir a un escenario externo.	0.5		
14	Será posible configurar el desplazamiento al salir de un escenario, generando que el personaje camine automáticamente.	1		
15	Será posible gestionar los escenarios de dos dimensiones según los archivos de configuración.	5		
16	Será posible leer los archivos de configuración de escenario externos o internos, además de su distribución de elementos mediante un gestor de escenarios.	3		
17	Será posible asociar un gestor de escenarios a un nivel.	2		
18	Será posible crear personajes mediante un archivo de configuración.	2	Sprint 2	25.5
19	Será posible configurar la naturaleza del personaje (Aliado, Neutro, Enemigo).	0.5		
20	Será posible configurar si pertenece a un batallón o a una orden específica.	0.5		
21	Será posible configurar un nombre al personaje.	0.5		
22	Será posible configurar las imágenes asociadas al personaje.	0.5		

Tesis publicada con autorización del autor
No olvide citar esta tesis

23	Será posible configurar la posición lógica en el escenario del personaje, como su posición absoluta (en píxeles).	0.5		
24	Será posible configurar la velocidad de desplazamiento de un personaje.	0.5		
25	Será posible asociar un camino de desplazamiento al personaje, con un destino lógico y absoluto en el escenario.	1		
26	Será posible configurar la selección del personaje por el usuario.	0.5		
27	Será posible configurar la orientación cardinal del personaje en el escenario.	0.5		
28	Será posible configurar el alcance de un proyectil lanzado por el personaje mediante una distancia absoluta (píxeles).	0.5		
29	Será posible configurar los puntos de magia (mana) de los personajes	0.5		
30	Será posible configurar los puntos de experiencia del personaje y un coeficiente de experiencia.	0.5		
31	Será posible configurar el nivel del personaje, basado en los puntos de experiencia y un coeficiente de experiencia que le permite pasar al siguiente nivel.	0.5		
32	Será posible configurar el estado del personaje (Respirando, Caminando, Atacando, Muerto o Curando)	0.5		
33	Será posible configurar las acciones que tiene el personaje (Atacar, Curar, Reclutar, etc) y las acciones que tiene activas en cierto momento.	1		
34	Será posible configurar la identificación del personaje con el que se está interactuando (Atacando, curando, etc.).	0.5		
35	Será posible configurar la zona de patrullaje del personaje en el escenario.	1		
36	Será posible configurar el nivel de juego al que pertenece el personaje (escenario externo).	0.5		
37	Será posible configurar el subnivel de juego al que pertenece el personaje (escenario	0.5		

Tesis publicada con autorización del autor
No olvide citar esta tesis

	interno).			
38	Será posible configurar los puntos de vida, ataque, manutención y defensa del personaje. Siendo manutención el precio que cuesta mantener el personaje en tu bando.	1		
39	Será posible configurar si el personaje es guardable, es decir si se guardará para futuros niveles de juego.	0.5		
40	Será posible configurar un radio de visibilidad en el escenario en valores absolutos que le permitirá interactuar con los otros personajes y eventos.	0.5		
41	Será posible crear diferentes tipos de personajes con diferentes cualidades que heredan de un personaje normal (Héroe, Curandero, Guerrero, etc.)	2		
42	Será posible configurar características especiales asociadas a diferentes tipos de personajes, como puntos de inteligencia, respeto y dinero pertenecientes al tipo Héroe.	0.5		
43	Será posible que un personaje acceda a los escenarios.			
44	Será posible controlar los atributos de los personajes en medio de la ejecución del videojuego			
45	Será posible controlar a más de un personaje a la vez.	1		
46	Será posible configurar un inventario asociado a un personaje, que contenga objetos de uso para el personaje e interacción con eventos.	1		
47	Será posible gestionar el cambio de nivel de experiencia de un personaje.	1		
48	Será posible leer los archivos de configuración de personajes asociados a un nivel en específico.	3		
49	Será posible gestionar los personajes mediante bandos (Aliando, Neutral, Enemigo)	2		
50	Será posible gestionar las misiones asociados a un nivel mediante un gestor de misiones o objetivos.	4	Sprint 3	22.5

Tesis publicada con autorización del autor
No olvide citar esta tesis

51	Será posible crear un archivo de configuración de misiones asociado a un nivel, como su estado de activación y su estado de visibilidad.	1		
52	Será posible crear un archivo de configuración de una misión que contendrá eventos asociados.	1		
53	Sera posible gestionar los eventos de una misión a través de archivos de eventos y configurar una descripción del nivel.	5		
54	Será posible crear un archivo de configuración de un evento.	1		
55	Será posible configurar el tipo de evento y el nombre del archivo de configuración de especificación de evento (Conversación, Decisión, Compra-Venta)	0.5		
56	Será posible configurar el nivel de juego y el subnivel de juego asociado al evento.	0.5		
57	Será posible configurar la alteración en puntos (Ataque, Defensa, Vida, etc) asociado al personaje que activa el evento.	0.5		
58	Será posible configurar los eventos a los que esta asociado el evento en cuestión. Además los personajes y objetos relacionados a la activación del evento.	0.5		
59	Será posible configurar la descripción del evento.	0.5		
60	Será posible crear archivos de especificación de evento de conversación.	1		
61	Será posible configurar la conversación, como los personajes que intervienen.	0.5		
62	Será posible controlar eventos de conversación.	2		
63	Será posible crear archivos de especificación de evento de decisión.	1		
64	Será posible configurar el número de decisiones asociadas al evento, las conversaciones y los atributos que alteran tomar la decisión, además los personajes asociados.	0.5		
65	Será posible controlar eventos de decisión.	3		
66	Será posible crear archivos de configuración de especificación de evento de compra-venta.	1	Sprint 4	23

Tesis publicada con autorización del autor
No olvide citar esta tesis

67	Será posible configurar la tienda asociada al evento compra-venta.	0.5		
68	Será posible crear archivos que representen el inventario de la tienda.	1		
69	Será posible controlar eventos de compra-venta	3		
70	Será posible crear archivos de configuración de especificación de evento de cambio de bando.	2		
71	Será posible configurar la zona de conversión de bando o el personaje en cuestión, como el bando origen y destino.	0.5		
72	Será posible controlar eventos de conversión de bando.	3		
73	Será posible configurar y controlar eventos que midan atributos para activar ciertos eventos.	2		
74	Será posible configurar y controlar eventos para que finalicen un nivel.	2		
75	Será posible configurar y controlar eventos que se activen al obtener cierto objeto.	2		
76	Será posible configurar y controlar eventos que verifiquen que se ha eliminado uno o más personajes de un batallón.	2		
77	Será posible configurar y controlar eventos que midan si un personaje está al borde de la muerte.	2		
78	Será posible configurar y controlar eventos que finalicen una misión específica.	2		
79	Será posible configurar y controlar eventos que activen un acceso a un escenario interno.	2		
80	Será posible configurar y controlar eventos que actualicen información de un evento que pertenece a la misión en desarrollo.	2		
81	Será posible configurar y controlar eventos que activen una nueva misión.	2	Sprint 5	21
82	Será posible configurar y controlar eventos que actualicen información de un evento que no pertenece a la misión en cuestión.	2		
83	Será posible configurar y controlar eventos que actualicen una misión con información de	2		

Tesis publicada con autorización del autor
No olvide citar esta tesis

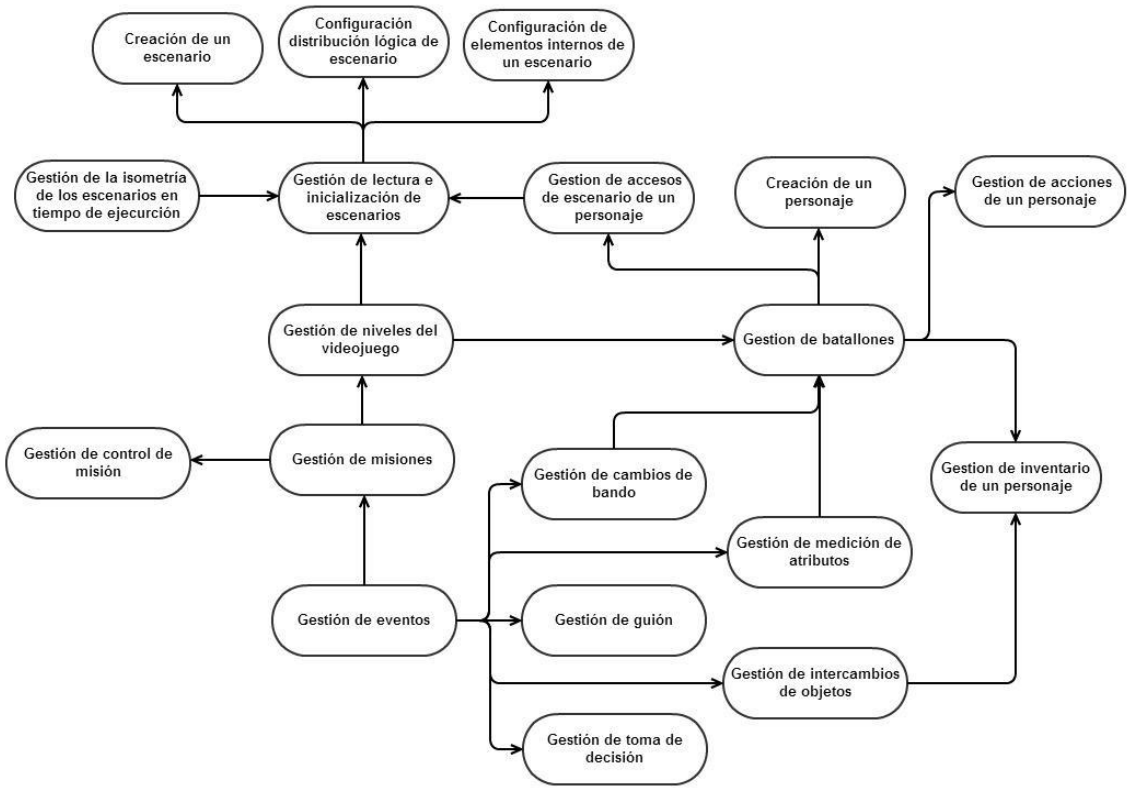
TESIS PUCP

	culminación.			
84	Será posible gestionar los eventos de manera concurrente según la configuración de misiones.	5		
85	Será posible gestionar la secuencia principal de juego según la gestión de niveles, escenarios y misiones.	6		

Tesis publicada con autorización del autor
No olvide citar esta tesis

2. Anexo 2: Mapa de proyecto

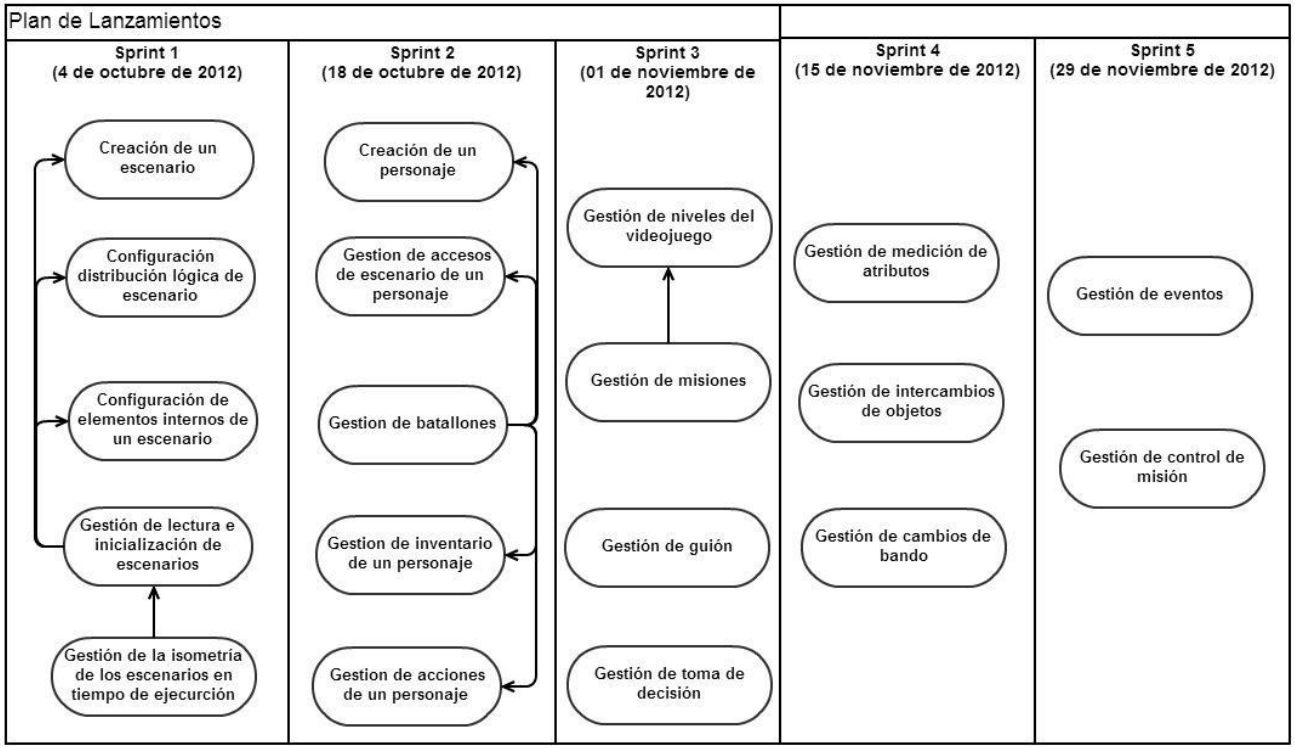
El mapa del proyecto es una representación gráfica de la distribución del trabajo, cuenta con dependencias para un mejor entendimiento del mismo.



Tesis publicada con autorización del autor
No olvide citar esta tesis

3. Anexo 3: Plan de lanzamientos

El plan de lanzamientos indica las fechas en las que se realizarán las entregas o lanzamientos del software. A partir de las historias de usuario y el mapa de proyecto se ha ordenado de acuerdo al catálogo de Sprints y las dependencias establecidas en el mapa de proyecto.



Tesis publicada con autorización del autor
No olvide citar esta tesis

TESIS PUCP

4. Anexo 4: Catálogo de riesgos

El catálogo de riesgos incluye una lista de los riesgos más importantes del proyecto, cada uno cuenta con una probabilidad y un tamaño de pérdida estimada en semanas. La exposición al riesgo es el producto de los dos valores mencionados y nos ofrece una prioridad de mitigación.

Código	Riesgo	Probabilidad Estimada	Tamaño de la pérdida (Semanas)	Exposición al riesgo
1	La curva de aprendizaje de la tecnología usada es más empinada de lo planeado	20%	4	0.8
2	El alcance del proyecto requiere mayor tiempo de desarrollo	20%	3	0.6
3	El desarrollo de una funcionalidad toma más tiempo de lo esperado	50%	1-4	0.5–2
4	El calendario de desarrollo estimado no está distribuido apropiadamente	25%	2	0.5
5	Falta de documentación del lenguaje de programación	10%	4	0.4
6	Cambios en la lista de exigencias/ historias de usuario.	30%	1	0.3
7	Falta incluir requerimientos necesarios	20%	1-4	0.2–0.8
8	Los dispositivos de desarrollo no cuentan con las herramientas necesarias para la implementación.	10%	2	0.2
9	Se pierde la información por robo o falla en funcionamiento del almacenamiento	5%	8-10	0.4 – 0.5
	Retraso estimado			**3.9–7.2**

Se han planteado medidas para mitigar, controlar o prevenir los riesgos mencionados en el catálogo de riesgos (Tabla 7.2). Estas medidas se exponen en la siguiente tabla.

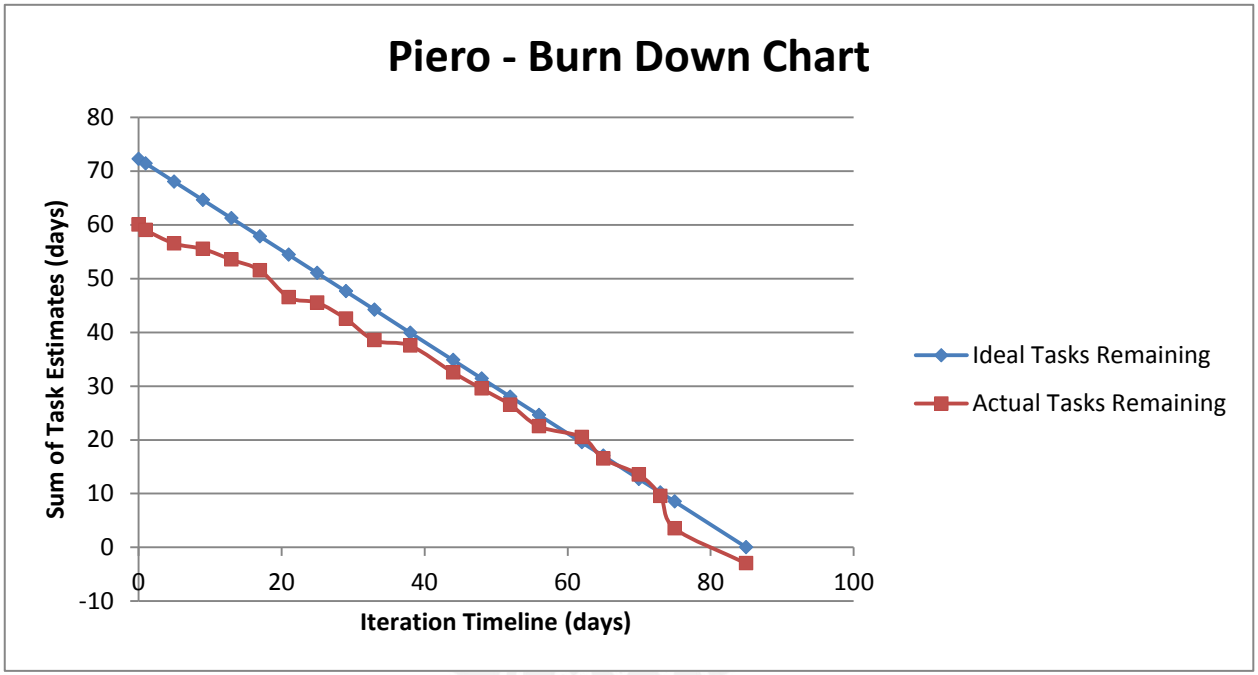
Tesis publicada con autorización del autor
No olvide citar esta tesis

Riesgo	Medida de control del riesgo
La curva de aprendizaje de la tecnología usada es más empinada de lo planeado	Asesoramiento en los temas pertinentes.
El alcance del proyecto requiere mayor tiempo de desarrollo	Evaluar alcance de proyecto. Limitar exigencias.
El desarrollo de una funcionalidad toma más tiempo de lo esperado	Desarrollar funcionalidades de mayor prioridad y postergar las de menor prioridad (Dentro de un mismo Sprint).
El calendario de desarrollo estimado no está distribuido apropiadamente	Redistribuir el contenido de desarrollo en los Sprints. Revaluar puntaje de prioridad de cada una de las historias de usuario.
Falta de documentación del lenguaje de programación	Adquirir documentación guía.
Cambios en la lista de exigencias/ historias de usuario.	Negociar el alcance. Revaluar la distribución de Sprints.
Falta incluir requerimientos necesarios	Al utilizar un ciclo de desarrollo incremental, se podrán añadir las nuevas funcionalidades. Previamente se debe haber revaluado la distribución de Sprints.
Los dispositivos de desarrollo no cuentan con las herramientas necesarias para la implementación.	Adecuada gestión de la configuración en caso de pérdida de equipo o mal funcionamiento del mismo.
Se pierde la información por robo o falla en funcionamiento del almacenamiento	Utilizar repositorio en un servidor externo. Mantener múltiples copias locales.

Tesis publicada con autorización del autor
No olvide citar esta tesis

5. Anexo 5: Burn Down Chart

Por medio de un diagrama Burn Down Chart se muestra el avance a lo largo del proyecto.



Tesis publicada con autorización del autor
No olvide citar esta tesis

6. Anexo 6: Plan de pruebas

Se utilizó la librería JUNIT para realizar las pruebas automatizadas. Estas se aplicaron a las clases pertinentes y su especificación se encuentra en el código del proyecto en la subcarpeta llamada *Test Packages*. A continuación se presenta el nombre de cada una de las pruebas, la funcionalidad asociada, el resultado esperado y el resultado obtenido.

Nombre de la Prueba	Nombre de la funcionalidad	Descripción	Resultado Esperado	Resultado Obtenido
Gestor de Nivel – Gestor de escenarios				**Aceptada**
testCargaConfcambioNivel()	cargaConfcambioNivel()	Configura el nuevo nivel al termino del anterior	Carga de nuevo nivel exitoso.	Aceptada
testCargaMisiones()	cargaMisiones()	Configura las misiones por nivel	Carga de misiones exitoso	Aceptada
testCargaSubNivel()	cargaSubNivel()	Configura los subniveles por cada nivel	Carga de subniveles exitoso	Aceptada
testSetCelda()	setCelda()	Se edita el contenido de una celda especifica del escenario	Edición exitosa	Aceptada
testGetCelda()	getCelda()	Se obtiene el contenido de una celda especifica del mapa	Obtención exitosa	Aceptada
testActualizaMapa()	actualizaMapa()	Se actualiza el escenario con los nuevos estados de celda	Actualización exitosa	Aceptada
testGetNiveles()	getNiveles()	Se obtienen los niveles del juego	Obtención exitosa	Aceptada
testSetNiveles	setNiveles()	Se establece el conjunto de niveles	Ajuste exitoso	Aceptada
testDevuelveMapaActual()	devuelveMapaActual()	Se obtiene el escenario actual	Obtención exitosa	Aceptada
testDevuelveNivelActual()	devuelveNivelActual()	Se obtiene el subnivel actual	Obtención exitosa	Aceptada
testDevuelveNivelPrincipalActual()	devuelveNivelPrincipalActual()	Se obtiene el nivel actual	Obtención exitosa	Aceptada

Tesis publicada con autorización del autor
No olvide citar esta tesis

testCargaNivel()	cargaNivel()	Se configura un nivel de juego	Configuración exitosa	Aceptada
testCargaElementos()	cargaElementos()	Se configuran los elementos internos de un nivel o subnivel del juego	Configuración exitosa	Aceptada
Gestor de Eventos				**Aceptada**
testGetEstadoEvento()	getEstadoEvento()	Se obtiene el estado del evento	Obtención exitosa	Aceptada
testSetEstadoEvento()	setEstadoEvento()	Se establece el estado del evento	Ajuste exitoso	Aceptada
testGetSigEvento()	getSigEvento()	Se obtiene el siguiente evento al actual	Obtención exitosa	Aceptada
testSetSigEvento()	setSigEvento()	Se establece el siguiente evento al actual	Ajuste exitoso	Aceptada
testGetTipoEvento()	getTipoEvento()	Se obtiene el tipo de evento	Obtención exitosa	Aceptada
testSetTipoEvento()	setTipoEvento()	Se establece el tipo de evento	Ajuste exitoso	Aceptada
testGetCeldaInicial()	getCeldaInicial()	Se obtiene la celda inicial del evento	Obtención exitosa	Aceptada
testSetCeldaInicial()	setCeldaInicial()	Se establece la celda inicial del evento	Ajuste exitoso	Aceptada
testGetDescripcion()	getDescripcion()	Se obtiene la descripción del evento	Obtención exitosa	Aceptada
testSetDescripcion()	setDescripcion()	Se establece la descripción del evento	Ajuste exitoso	Aceptada
testGetObjetosPremio()	setObjetosPremio()	Se establecen los objetos que se podrán ganar al pasar el evento	Ajuste exitoso	Aceptada
testSetObjetosPremio()	getObjetosPremio()	Se obtienen los objetos que se podrán ganar al pasar el evento	Obtención exitosa	Aceptada
testGetEventosRelacionados()	getEventosRelacionados()	Se obtienen los eventos relacionados al evento para su ejecución.	Obtención exitosa	Aceptada
testSetEventosRelacionados()	setEventosRelacionados()	Se establecen los eventos relacionados al evento para su ejecución.	Ajuste exitoso	Aceptada
testGetPersonajesRelacionados()	getPersonajesRelacionados()	Se obtienen los personajes relacionados al evento para su creación.	Obtención exitosa	Aceptada
testSetPersonajesRelacionados()	setPersonajesRelacionados()	Se establecen los personajes relacionados al evento para su creación.	Ajuste exitoso	Aceptada

Tesis publicada con autorización del autor
No olvide citar esta tesis

testLeePersonaje()	leePersonaje()	Se leen el personaje asociado para su creación.	Lectura exitosa	Aceptada
testGetAtributos()	getAtributos()	Se obtienen los atributos que afectan al personaje principal al pasar el evento.	Obtención exitosa	Aceptada
testSetAtributos()	setAtributos()	Se establecen los atributos que afectan al personaje principal al pasar el evento.	Ajuste exitoso	Aceptada
testGetDireccion()	getDireccion()	Se obtiene la dirección del evento.	Obtención exitosa	Aceptada
testSetDireccion()	setDireccion()	Se establece la dirección del evento.	Ajuste exitoso	Aceptada
testGetPersonajeActivacion()	getPersonajeActivacion()	Se obtiene el personaje que activa el evento	Obtención exitosa	Aceptada
testSetPersonajeActivacion()	setPersonajeActivacion()	Se establece el personaje que activa el evento	Ajuste exitoso	Aceptada
testGetNivelCambio()	getNivelCambio()	Se obtiene el nivel al que se cambia al pasar el evento	Obtención exitosa	Aceptada
testSetNivelCambio()	setNivelCambio()	Se establece el nivel al que se cambia al pasar el evento.	Ajuste exitoso	Aceptada
Gestor de Personajes				**Aceptada**
testGetStrBatallon()	getStrBatallon()	Se obtiene el nombre del batallón	Obtención exitosa	Aceptada
testSetStrBatallon()	setStrBatallon()	Se establece el nombre del batallón	Ajuste exitoso	Aceptada
testGetStrLider()	getStrLider()	Se obtiene el nombre del personaje líder del batallón	Obtención exitosa	Aceptada
testSetStrLider()	setStrLider()	Se establece el nombre del personaje líder del batallón	Ajuste exitoso	Aceptada
testInvocarApus()	invocarApus()	Se alteran las estructuras internas (atributos, personaje selección) para la ejecución del poder.	Ejecución exitosa	Aceptada
testEjecutarinvocarApus()	ejecutarinvocarApus()	Se ejecuta el poder característico de un personaje.	Ejecución exitosa	Aceptada
testAgregaObjeto()	agregaObjeto()	Se agregan objetos al inventario de un personaje.	Ejecución exitosa	Aceptada
testGetInventario()	getInventario()	Se obtienen el inventario de un personaje.	Obtención exitosa	Aceptada

Tesis publicada con autorización del autor
No olvide citar esta tesis

testSetInventario()	setInventario()	Se establece el inventario de un personaje.	Ajuste exitoso	Aceptada
testUsarObjeto()	usarObjeto()	Se usa un objeto del inventario de un personaje.	Ejecución exitosa	Aceptada
testGetInteligencia()	getInteligencia()	Se obtiene los puntos de inteligencia de un personaje	Obtención exitosa	Aceptada
testSetInteligencia()	setInteligencia()	Se establecen los puntos de inteligencia de un personaje	Ajuste exitoso	Aceptada
testGetDinero()	getDinero()	Se obtiene los puntos de dinero de un personaje	Obtención exitosa	Aceptada
testSetDinero()	setDinero()	Se establece los puntos de dinero de un personaje	Ajuste exitoso	Aceptada
testGetRespeto()	getRespeto()	Se obtiene los puntos de respeto de un personaje	Obtención exitosa	Aceptada
testSetRespeto()	setRespeto()	Se establece los puntos de respeto de un personaje	Ajuste exitoso	Aceptada
testHayEspacio()	hayEspacio()	Se verifica si el personaje aún tiene espacio en su inventario.	Ejecución exitosa	Aceptada
testCurar()	curar()	Se alteran los puntos de vida de la unidad curada y se restan los puntos de magia de la curadora.	Ejecución exitosa	Aceptada
testGetVidaMax()	getVidaMax()	Se obtiene la vida máxima de un personaje	Obtención exitosa	Aceptada
testSetVidaMax()	setVidaMax()	Se establece la vida máxima de un personaje	Ajuste exitoso	Aceptada
testGetVidaAct()	getVidaAct()	Se obtiene la vida actual de un personaje	Obtención exitosa	Aceptada
testSetVidaAct()	setVidaAct()	Se establece la vida actual de un personaje	Ajuste exitoso	Aceptada
testGetAtaque()	getAtaque()	Se obtiene los puntos de ataque de un personaje	Obtención exitosa	Aceptada
testSetAtaque()	setAtaque()	Se establece los puntos de ataque de un personaje	Ajuste exitoso	Aceptada
testGetDefensa()	getDefensa()	Se obtiene los puntos de defensa de un personaje	Obtención exitosa	Aceptada
testSetDefensa()	setDefensa()	Se establece los puntos de defensa de un personaje	Ajuste exitoso	Aceptada

Tesis publicada con autorización del autor
No olvide citar esta tesis

testGetNombre()	getNombre()	Se obtiene el nombre de un personaje	Obtención exitosa	Aceptada
testSetNombre()	setNombre()	Se establece el nombre de un personaje	Ajuste exitoso	Aceptada
testGetRadioVisibilidad()	getRadioVisibilidad()	Se obtiene el radio de visibilidad de un personaje	Obtención exitosa	Aceptada
testSetRadioVisibilidad()	setRadioVisibilidad()	Se establece el radio de visibilidad de un personaje	Ajuste exitoso	Aceptada
testGetX()	getX()	Se obtiene la posición en la coordenada X con respecto a los pixeles de la interfaz.	Obtención exitosa	Aceptada
testSetX()	setX()	Se establece la posición en la coordenada X con respecto a los pixeles de la interfaz.	Ajuste exitoso	Aceptada
testGetPosLogicaX()	getPosLogicaX()	Se obtiene la posición en la coordenada lógica X con respecto a la distribución isométrica del escenario.	Obtención exitosa	Aceptada
testSetPosLogicaX()	setPosLogicaX()	Se establece la posición en la coordenada lógica X con respecto a la distribución isométrica del escenario.	Ajuste exitoso	Aceptada
testIsSeleccionado()	isSeleccionado()	Se verifica si el personaje esta seleccionado.	Ejecución exitosa	Aceptada
testGetDireccion()	getDireccion()	Se obtienen la dirección cartesiana del personaje	Obtención exitosa	Aceptada
testSetDireccion()	setDireccion()	Se establece la dirección cartesiana del personaje.	Ajuste exitoso	Aceptada
testGetVelocidad()	getVelocidad()	Se obtiene la velocidad de movimiento del personaje	Obtención exitosa	Aceptada
testGetReclutable()	getReclutable()	Se obtiene la capacidad de un personaje de ser reclutable	Obtención exitosa	Aceptada
testSetReclutable()	setReclutable()	Se establece la capacidad de un personaje de ser reclutable	Ajuste exitoso	Aceptada
testGetExperiencia()	getExperiencia()	Se obtiene los puntos de experiencia de un personaje	Obtención exitosa	Aceptada
testSetExperiencia()	setExperiencia()	Se establece los puntos de experiencia de un personaje	Ajuste exitoso	Aceptada
testGetMana()	getMana()	Se obtiene los puntos de magia de un personaje	Obtención exitosa	Aceptada

Tesis publicada con autorización del autor
No olvide citar esta tesis

testSetMana()	setMana()	Se establece los puntos de magia de un personaje	Ajuste exitoso	Aceptada
testGetBatallon()	getBatallon()	Se obtiene el batallón al que pertenece el personaje	Obtención exitosa	Aceptada
testSetBatallon()	setBatallon()	Se establece el batallón al que pertenece el personaje	Ajuste exitoso	Aceptada
Gestor de la secuencia principal de juego				**Pendiente**
testVerificaReclutamiento()	verificaReclutamiento()	Se verifica que el reclutamiento se lleve a cabo de manera correcta. Cambio de nivel, actualización de atributos.	Ejecución exitosa	Aceptada
testEjecutarFinal()	ejecutarFinal()	Se ejecuta el final de un nivel y se reciclan las estructuras según la configuración del próximo nivel.	Ejecución exitosa	Pendiente
testEjecutarDecision()	ejecutarDecision()	Ejecuta una decisión ante un evento de la misma naturaleza, afectando las estructuras pertinentes.	Ejecución exitosa	Aceptada
testEjecutarCambio()	ejecutarCambio()	Ejecuta un cambio de bando de un personaje ante un evento de la misma naturaleza, afectando las estructuras pertinentes.	Ejecución exitosa	Aceptada
testVerificaEvento()	verificaEvento()	Verifica la zona de activación del evento o el personaje de activación del mismo	Ejecución exitosa	Aceptada
testRestauraMana()	restauraMana()	Se restauran los puntos de magia de los personajes que poseen dicha característica.	Ejecución exitosa	Aceptada
testRevisaAcceso()	revisaAcceso()	Se revisan los accesos de cada nivel y subnivel en tiempo de ejecución.	Ejecución exitosa	Pendiente
testEjecutaCambioNivel()	ejecutaCambioNivel()	Ejecuta un cambio de nivel, con la inicialización de las estructuras del nuevo nivel.	Ejecución exitosa	Pendiente
testEjecutaGuion()	ejecutaGuion()	Ejecuta un guión ante un evento de la misma naturaleza, afectando las estructuras pertinentes.	Ejecución exitosa	Aceptada
testBuscaEvento()	buscaEvento()	Se busca la activación de un evento en tiempo de ejecución, como respuesta de las entradas del usuario.	Ejecución exitosa	Aceptada

Tesis publicada con autorización del autor
No olvide citar esta tesis

TESIS PUCP

testEjecutaEvento()	ejecutaEvento()	Se ejecutan eventos de menor jerarquía y que no tienen una funcionalidad separada como la de guión o decisión.	Ejecución exitosa	Aceptada

Tesis publicada con autorización del autor
No olvide citar esta tesis