UNIVERSIDAD NACIONAL DE LA PLATA

FACULTAD DE INFORMÁTICA

TESINA DE LICENCIATURA

Título: Monitoreo y Sistema de recuperación en Juegos Educativos Móviles
Autores: Sebastián Andrés Perez Escribano
Director: Dra. Silvia Gordillo
Codirector: Dra. Cecilia Challiol
Asesor profesional:
Carrera: Licenciatura en Informática

Resumen

Las Aplicaciones Móviles fueron en un principio pensadas para aumentar la productividad: correo electrónico, calendario, base de datos de contactos, etc. Pero gracias al auge de estas y a las mejoras en el hardware, las Aplicaciones Móviles se esparcieron en otras áreas como Servicios de geo-localización, Juegos Móviles entre otras. Para este trabajo nos interesan aquellas Aplicaciones Móviles basadas en posicionamiento. En esta clase de aplicación el usuario recibe información diferente en cada posición acorde a la naturaleza de la aplicación .El seguimiento en tiempo real del alumno en los Juegos Educativos Móviles basados en posicionamiento es fundamental, y es un campo abierto de investigación. En esta tesis se presenta una herramienta que permite a los docentes monitorear este tipo de juegos. Con esta herramienta creada el docente puede observar el comportamiento de los alumnos sin estar necesariamente presente en el lugar. Además, esta herramienta permite guiar a los alumnos, por ejemplo, mediante la idea de docentes tutores. Una problemática de los Juegos Educativos Móviles basados en posicionamiento, son los problemas técnicos que puedan surgir por el uso de los dispositivos móviles pérdida de señal de WiFi, problemas con el SO, se reinicia el dispositivo móvil, entre otros. Con la herramienta creada se proporcionan una serie de soluciones a estos problemas.

Palabras Claves

Computación Móvil. Juegos Educativos Móviles. Aplicaciones basadas en el contexto. Aplicaciones basadas en posicionamiento. Aplicación Web.

Conclusiones

Se presentó un modelo que muestra las características básicas de los Juegos Educativos Móviles basados en posicionamiento.
Se profundizó sobre el rol que cumplen los docentes en estos juegos en particular.
Mediante la implementación del modelo propuesto, se pudieron descubrir problemáticas específicas sobre estas aplicaciones.

Trabajos Realizados

Para el desarrollo del presente trabajo de tesis primero se realizó un modelo de los aspectos mínimos para tener una aplicación de las características presentadas en los Juegos Educativos Móviles basados en posicionamiento. Se generó la herramienta para monitoreo y recuperación en Juegos Educativos Móviles basados en posicionamiento y se realizaron algunas pruebas prototípicas con la dicha herramienta.

Trabajos Futuros

La herramienta presenta varios puntos en los cuales se puede enriquecer, tales como por ejemplo: agregar un chat entre los alumnos y los docentes; agregar una vía de comunicación entre docentes y tutores (de tipo chat); agregar el muestreo de datos estadísticos de forma clara y concisa; crear un editor para crear y modificar los juegos de forma fácil.
Por otro lado es fundamental probar la herramienta con grupos masivos de gente y con juegos con una complejidad mayor a la planteada en este trabajo para enriquecer a partir de estos resultados la herramienta

Fecha de la presentación: Abril del 2013

Índice general

Índice de figuras

Índice de cuadros

Agradecimientos

Quiero agradecer en principio a mi familia que me dio todo para poder llegar a donde estoy, a mi padre Juan José que me ha llevado siempre a la facultad. A mi madre Susana que me dio su tiempo para escucharme siempre. A mi hermana Gabriela, que está siempre conmigo dándome todo su apoyo incondicional.

En el ámbito académico quisiera agradecer a Gustavo, que me abrió las puertas del LIFIA. A Silvia por brindarme su apoyo como directora de este trabajo y siempre tener confianza en mi. A Cecilia, por ayudarme gratamente y brindándome su tiempo para resolver diferentes problemas. También quisiera agradecer especialmente a Andrés, quien me sirvió como un gran referente de lo que es ser un investigador incansable, un visionario y por sobre todo una buena persona. A todos los compañeros con quienes he tenido el placer de compartir grandes momentos en el laboratorio, Juli, Nacu, Santi, Lautaro, Ale, Nati, Diego, Mati, Keko, Nacho y Lucas, Juani, Gustavo, 'Los Juanes', entre otros.

A mis compañeros de la facultad. Aquellos con quienes compartimos este camino a lo largo de tantos años. A Facu, Noe, Ceci, el colo. A Emiliano, con quien además de encontrar un compañero de estudio, encontré a un amigo.

A mis compañeros y profes de entrenamiento con me quienes formé a lo largo de toda mi vida. A Rolo, Cristian, Diego y Javi, entre otros.

Y por último, pero no por ello menos importantes, a mis amigos de toda la vida. Aquellos con quienes quiero seguir compartiendo mi vida. A Emi, Fer y Rorro.

Capítulo 1

Introducción

1.1. Juegos Educativos Móviles

La amplia evolución de los dispositivos móviles ha hecho que sean instrumentos utilizados en diferentes dominios y se ha ido integrado como herramientas cotidianas. El crecimiento de aplicaciones creadas sobre estos dispositivos ha aumentado en los últimos años de manera ininterrumpida haciendo que sea un fenómeno en la industria del software. Las aplicaciones que corren sobre un dispositivo móvil se conocen como Aplicaciones Móviles.

Actualmente, las prestaciones que presentan los dispositivos móviles sobre todo las mejoras en el hardware, hacen que la gama de Aplicaciones Móviles sea muy variadas. Entre las prestaciones se pueden mencionar, por ejemplo, acceso permanente a Internet (3G, WIFI), sistemas de posicionamiento (GPS, WPS), detector de movimiento (acelerómetro), entre otros.

Las Aplicaciones Móviles fueron en un principio pensadas para aumentar la productividad: correo electrónico, calendario, base de datos de contactos, entre otros. Pero gracias al auge de estas y a las mejoras en el hardware, las Aplicaciones Móviles se esparcieron en otras áreas como Servicios de geo-localización, Juegos Móviles entre otras.

Para este trabajo nos interesan aquellas Aplicaciones Móviles basadas en posicionamiento. En esta clase de aplicación el usuario recibe información diferente

en cada posición acorde a la naturaleza de la aplicación. Un ejemplo típico son las guías turísticas (por ejemplo, Metro Paris Phone [18]), donde se brinda, por ejemplo, información turística al usuario dependiendo dónde se encuentran, mapas para determinar su posición actual y la posibilidad de realizar búsquedas caminos para poder llegar a puntos de interés turísticos.

Existe una amplia gama de Aplicaciones Móviles basadas en posicionamiento, una clase particular son los Juegos Móviles (basados en posicionamiento). Estos juegos se utilizan con distintos fines, por ejemplo: Entretenimiento (Monopoly [14], GAMMA [11], Pirates! [4]), Actividad Física (PiNiZoRo[26]) o Educativos (Savannah [3], Frequency 1550 [20], MobileMath [27], HasletInteractive [10]).

En particular, en este trabajo nos focalizaremos en Juegos Educativos Móviles basados en posicionamiento. A continuación se presentan distintos ejemplos de este tipo de juego para comprender mejor su funcionamiento.

- *Savannah* [3]: Juego en el que los jugadores interpretan cada uno a un león que se encuentra con su manada dentro de una sabana en la cual deben sobrevivir. Para ello deben recorrer el lugar el cual es un espacio abierto y encontrar tanto agua como animales para cazar para sobrevivir. Al momento de la caza, los jugadores deben coordinar sus movimientos para que la misma sea un éxito. Si pueden cazar a la presa entonces pueden alimentarse y seguir viviendo, caso contrario la energía de cada jugador se ve disminuida. Si en algún momento del juego alguno de los jugadores se queda sin energía debe dirigirse a un espacio particular donde debe quedarse un período de tiempo en el cual se lo revive para poder reingresar al juego. La característica fundamental de este juego es la colaboración de los integrantes para elegir las presas y planear como realizar el ataque ya que no todos los ataques serán efectivos. Esta colaboración debe hacerse fuera del sistema, es decir de manera real al igual que el desplazamiento dentro del campo de juego.

- *Frecuency 1550* [20]: Juego donde los alumnos deben recorrer la ciudad de Amsterdam como si fuera en el año 1550. Dentro del juego los alumnos aprenden sobre la vida en la época medieval holandesa y sobre los estratos sociales. La aplicación funciona como excusa para que el aprendizaje sea más llevadero e incentive al alumno a aprender. Durante el juego se suceden varios eventos vinculados a un crimen el cual los alumnos deben resolver. Esta aplicación ejemplifica cómo utilizando una historia de misterio los alumnos se ven más involucrados en los hechos históricos y a diferentes temas explicados en clase.

- *MobileMath* [27]: Juego donde los alumnos deben conformar equipos y recorrer la ciudad de Utrech Holanda. Para ello se les brinda un dispositivo

móvil con GPS con el cual deberán caminar y conformar diferentes figuras geométricas. Mientras más figuras realicen más puntos obtendrán, el equipo que tenga más puntos pasado cierto tiempo gana. Aquí la idea subyacente es la de utilizar los conceptos de geometría vistos en clase para poder ganar el juego. También fomenta el trabajo en equipo.

- *HasletInteractive* [10]: Juego en el cual se presenta un proyecto donde grupos de chicos de escuelas primarias participan de una historia de misterio en la cual aprenderán temas de la naturaleza (tales como biología, geografía y matemáticas) a través de trabajo de campo. El proyecto tuvo lugar en Aarhous Dinamarca y la historia en la cual se sumergen los alumnos es en un futuro donde un poderoso virus afectó a la naturaleza. Los alumnos deben recorrer diferentes zonas dentro de un parque, tratando de aprender más sobre el virus y tratarán de restablecer el orden en la naturaleza. Para ello, deberán moverse dentro del parque y obtener datos, ya sea notas, fotos o videos. A su vez recibirán ayuda por parte del sistema como notas o ayudas audiovisuales en caso que sean necesarias. Este proyecto toma varios elementos importantes como el trabajo en equipo, el uso de los dispositivos móviles como elemento para tomar muestras de datos, capacidad de aplicar los conocimientos aprendidos en clase en un entorno real, entre otros aspectos.

Como se pudo apreciar las características de los Juegos Educativos Móviles basados en posicionamiento son muy variadas, por ejemplo, pueden poseen ciertas características como, capacidad de conectarse a otros dispositivos para intercambiar información tanto entre alumnos como entre los profesores, conexión permanente a Internet para acceder a información cuando sea necesario, entre otros. Estos juegos intentan incluir la tecnología dentro del ámbito educativo para hacerlo más llamativo y motivador para el alumno. Como parte del juego se puede utilizar características del contexto (además de la posición del alumno, por ejemplo, qué lo rodea), para lograr más inmersión por parte de los alumnos en él mismo. La idea principal no es reemplazar por completo el modelo de enseñanza actual sino brindar más herramientas al docente para que el alumno se sienta más motivado a la hora de aprender.

Actualmente, la mayoría de estos juegos se crean ad-hoc mediante el uso de prototipos y se centran como se especificó en los juegos presentados (Savannah, Frequency 1550, MobileMath, HasletInteractive) en la perspectiva del alumno. Es decir, estos juegos se encuentran focalizados en el alumno, pero no existe la idea de un observador del juego (por ejemplo, un docente). Por lo tanto, aquel que no se encuentra participando no puede saber qué está ocurriendo (sólo puede observar el movimiento de los alumnos).

Un Juego Educativo Móvil que no está basado en posicionamiento y donde se tiene el rol del docente como observador, es el juego SMILE [25] (Standorfd Mobile Inquiry-Based Learning Environment)[1]. En este juego se tiene una herramienta que ayuda al docente a visualizar qué está haciendo cada grupo de manera pasiva, es decir que permite tener un control sobre la actividad del alumno. Es decir, se hace un seguimiento del alumno en tiempo real. Hay que tener en cuenta, que en el caso de SMILE los alumnos no se mueven del aula (el sistema no es basado en posicionamiento).

El seguimiento en tiempo real del alumno en los Juegos Educativos Móviles basados en posicionamiento es fundamental, y es un campo abierto de investigación. Esto es una de las motivaciones de esta tesis. Se necesita contar con una herramienta que le permita al docente hacer un monitoreo de todo el juego, ya que los alumnos en este tipo de juego están en constante movimiento. Con esta herramienta el docente podría observar el comportamiento de los alumnos sin estar necesariamente presente en el lugar y sin cohibir a los alumnos por la presencia del mismo. Además, esta herramienta podría permitir guiar a los alumnos, por ejemplo, mediante la idea de docentes tutores. En este tipo de juegos (Educativos Móviles basados en posicionamiento), existen muchos alumnos que están constantemente en movimiento, una herramienta como la descripta permitiría que varios docentes estén monitoreando a distintos alumnos en tiempo real.

Otra problemática de las Aplicaciones Móviles en general, y los Juegos Educativos Móviles basados en posicionamiento no son la excepción, son los problemas técnicos que puedan surgir por el uso de los dispositivos móviles. Algunos de los problemas pueden ser, pérdida de señal de WiFi, problemas con el SO, se reinicia el dispositivo móvil, entre otros. Todos estos problemas repercuten de manera importante en los juegos, ya que si el alumno no puede completar el mismo puede frustrarse y perder la motivación. Ante este panorama, los docentes siempre necesitan de un experto en tecnología para dar solución a estas problemáticas. Esto hace que la puesta en práctica de los juegos sea dependiente de contar con un experto tecnológico. Es decir, el docente no cuenta con las herramientas para solucionar los problemas que puedan surgir en el juego. Lo anteriormente descripto es otra de las motivaciones de esta tesis, proveer al docente de un sistema de recuperación de errores, esto permitirá que el docente cuente con las herramientas necesarias para poder hacer que ante un problema técnico el alumno pueda

[1]SMILE es un juego en el cual dentro de una clase los alumnos generan preguntas para que otros puedan responderlas. Los alumnos conforman grupos y cada grupo genera preguntas referentes al tema del día. Para ello utilizan los celulares o tablets para generar las preguntas y las respuestas, agregando de ser necesario alguna fotografía obtenida con el mismo dispositivo. Luego esa pregunta es evaluada por el profesor quien determina si puede ser o no efectuada al resto de los grupos y en caso de serlo, se distribuye. Cada grupo contesta las preguntas que se vayan generando y aquel que acierte más es el ganador.

seguir jugando, por ejemplo, usando otro dispositivo, siempre manteniendo la información del juego que tenia antes de que surgiera el problema.

1.2. Objetivos

El objetivo general de la tesis es crear una herramienta para el seguimiento (monitoreo) de los alumnos en el marco de los Juegos Educativos Móviles basados en posicionamiento. Esta herramienta está pensada para ser usada por los docentes, y debe ser fácil de usar sin la necesidad de tener mayores conocimientos de informática. El docente contará, en tiempo real, con la información de la actividad que está haciendo cada alumno como así también todo lo que ya hicieron.

Esta herramienta, además, permitirá que el docente la utilice para actuar como sistema de recuperación ante determinadas situaciones. En particular, se detallarán posibles problemáticas que podrían llegar a ocurrir en un juego y cómo el docente usando la herramienta puede hacer que el alumno continué su juego.

A continuación se describen los objetivos específicos del trabajo:

- Presentar los conceptos de los Juegos Educativos Móviles basados en posicionamiento, esto abarca:
  - Descripción de los elementos fundamentales de este tipo de juego.
  - Seguimiento de un juego de ejemplo para visualizar los elementos destacados, junto al comportamiento de los alumnos y las acciones que este puede realizar en estos juegos.
- Especificar un Modelo del Usuario mínimo para poder llevar a cabo una herramienta como la que se presenta en esta tesis. Es decir, lograr un seguimiento del alumno, y poder hacer recuperación de fallas.
- Enunciar las problemáticas en Juegos Educativos Móviles basados en posicionamiento. Esto permitirá poder determinar qué situaciones se deben tener en cuenta para crear un sistema de recuperación. Por un lado, se describirán algunas de las problemáticas que se pueden encontrar tanto a nivel de software como de hardware. Luego, se presentarán posibles soluciones a las problemáticas mencionadas.
- Presentar la Herramienta que permite hacer el seguimiento de los alumnos y además sirve como sistema de recuperación. La herramienta se ira explicando por parte, mostrando como usarla:

- Para hacer seguimiento de los alumnos basándose en el Modelo de Usuario presentado.
- Como sistema de recuperación, acorde a las problemáticas planteadas.

1.3. Estructura del trabajo

En este capítulo se han introducido los temas principales que cubre el trabajo y se han indicado los problemas que se atacaran, cómo se analizaran los mismos y el rumbo que tomará la solución propuesta. En el Capítulo 2 se presenta el marco sobre las Juegos Educativas Móviles basados en posicionamiento y los conceptos que luego en el Capítulo 3 se utilizarán para generar un modelo de estas aplicaciones y particularmente un el modelo para los usuarios. En el Capítulo 4 se mostrarán las problemáticas que poseen este tipo de aplicaciones junto con sus posibles soluciones. Luego en el Capítulo 5 se explicará la herramienta propuesta para resolver los problemas planteados. El trabajo se cierra en el Capítulo 6 con una conclusión final sobre los aportes presentados y el trabajo futuro.

Capítulo 2

Juegos Educativos Móviles

En este capítulo se mencionarán las características básicas de los Juegos Educativos Móviles basados en posicionamiento. En particular, se describirán los aspectos relevantes del juego y el ambiente en donde se llega a cabo como así también el comportamiento del alumno en este tipo de juego.

2.1. Esquema de juego

En la actualidad, existen varios Juegos Educativos Móviles con interacción activa de los alumnos, por ejemplo: HasleInteractive [15] y [20]. Estos juegos se basan en una historia la cual sirve como hilo conductor del juego. La historia ayuda a los alumnos a involucrarse en el juego, haciendo que la experiencia sea más entretenida y satisfactoria. Como se menciona en [5], una historia enmarca el contexto en el cual se plantea las actividades como así también los distintos roles que pueden desempeñar los alumnos.

La historia puede estar dividida en escenas, en cada escena los alumnos reciben, mediante el uso de dispositivos móviles, información del juego. La información puede abarcar, por ejemplo: pistas relacionadas al juego, preguntas que deben responder los alumnos, entre otros aspectos relacionados a la naturaleza del juego. Todas las acciones que realicen los alumnos deben estar enmarcadas acordes a la historia de base, para lograr así más inmersión por parte del alumno.

Si bien existen diferentes dinámicas de juegos como se mencionan en [13], en este trabajo nos focalizaremos en aquellos juegos que están planteados como un 'domino´. En [13] se plantea esta dinámica (de 'dominó') como un juego en donde el jugador estando en un lugar determinado puede elegir donde continuar. Es decir, las historias están planteadas como en los libros de Elige tu propia aventura. En este tipo de libros, el lector asume el rol de protagonista de la historia[1] y en ciertos momentos de la misma puede tomar decisiones dentro de la misma. Estas decisiones implicarán que el rumbo de la historia va cambiando con respecto a lo elegido por el lector. Es decir, el relato no sigue la linealidad tradicional de narrativa de un libro normal. De manera análoga, en los juegos con este tipo de dinámica ocurre lo mismo. Cada alumno puede optar hacia dónde ir y modificar el relato según sus decisiones. Por cada escena se pueden tener varias escenas siguientes, de las cuales el alumno deberá escoger una de ellas para continuar con la historia.

A partir de los conceptos nombrados podemos tener una idea simplista de como se conformarán los juegos mencionados. En la Figura 2.1 se muestra un ejemplo esquemático de la estructura de estos juegos. En este ejemplo, se puede observar que se cuenta con la historia de base que es el hilo conductor del juego y además una escena inicial, en la cual comienza la historia. Cada escena cuenta con varias escenas siguientes y existe al menos una escena final en la cual el juego termina. La figura hace foco en una de las escenas, para mostrar una de las preguntas que debe responder el alumno como parte del juego.

Para que este tipo de juegos puedan ser utilizados por los docentes para realizar evaluaciones de distintos tipo, es necesario que soporten la generación de cuestionarios (preguntas) y la evaluación de los mismos. Esta evaluación se podría realizar por ejemplo, de manera automática o requerir que un docente evalué cada pregunta. En este trabajo, nos focalizaremos en aquellos juegos que contienen preguntas que se pueden evaluar de manera automática. Esto permite tener una evaluación en tiempo real, y darle al alumno un resultado al finalizar el juego. Ante la necesidad de evaluar las repuestas dadas en tiempo real surge la necesidad de saber cuando una pregunta es correcta o incorrecta. En este trabajo, simplificaremos este aspecto asumiendo que una pregunta se conforma por el enunciado de la misma y sus respectivas respuestas, tanto sean correctas como incorrectas. De esta manera, podremos evaluar fácilmente si las respuestas que dan los alumnos son correctas o no, acorde a la respuesta que seleccionó el alumno. La Figura 2.2 amplia la pregunta destacada, agregándole las posibles respuestas. Se puede observar que se indica cuáles son las opciones correctas y las incorrectas.

[1]Para nuestro ejemplo esta historia cuenta con un detective que debe develar si la muerte de una persona fue intencional o por accidente

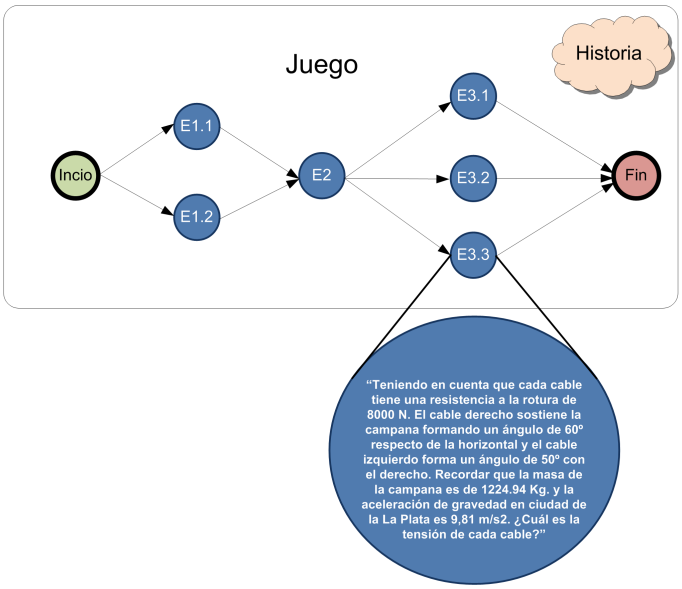

Figura 2.1: Esquema de juego

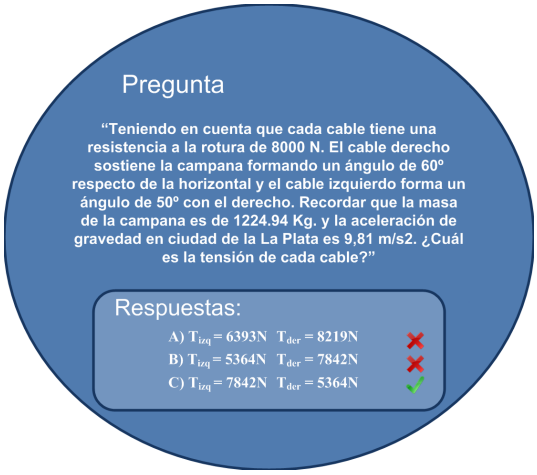

Figura 2.2: Pregunta de ejemplo.

Acorde a la naturaleza del juego, podría asignarse puntaje acorde a las respuestas que va brindando un alumno, tanto sea sumando en el caso de contestar correctamente como restando en el caso contrario. También se podría agregar al juego restricciones de tiempo, esto hará que, por ejemplo, la experiencia de juego sea en un tiempo determinado o que cada pregunta tenga un tiempo máximo para responderla. La variada gama de posibilidades sólo agrega complejidad al juego, por esta razón, en este trabajo se tendrán en cuenta juegos simples que sólo consisten en presentarles preguntas a los alumnos.

En [12] se nombran siete aspectos para promover la motivación y el aprendizaje. Veamos cómo son aplicados algunos de estos aspectos en los Juegos Educativos Móviles basados en posicionamiento que son de nuestro interés.

- Retos: Las preguntas que los alumnos deben responder para poder avanzar en el juego.
- Curiosidad: Resolver el misterio propuesto en la historia que enmarca al juego y que sirve como marco para realizar las diferentes evaluaciones.
- Control: El alumno tiene el control sobre el juego ya que mediante sus decisiones (de elección de la siguiente escena) el mismo irá variando.
- Fantasía: La historia en sí misma hace que los alumnos deban utilizar la imaginación para ponerse en contexto en el juego.

2.2. Aspectos de movilidad de los juegos

Dado que el marco de trabajo de esta tesis son los Juegos Educativos Móviles basados en posicionamiento, el ambiente físico (espacio físico o ambiente de juego) donde se lleva a cabo el juego cumple un rol fundamental. Por un lado, el ambiente físico determina el espacio en donde se llevará a cabo el juego, pero también determina donde se pueden posicionar las preguntas relacionadas al mismo.

Por una cuestión de flexibilidad, y acorde a lo planteado en [1], se eligió trabajar de manera desacoplada el juego del ambiente donde se realiza (juega) el mismo. De esta manera, el mismo juego (historia y escenas) puede ser realizado en distintos lugares, por ejemplo: plazas, parques, escuelas, entre otros.

Siguiendo con el ejemplo planteado en la Figura 2.1, definidos dos ambientes de juegos como se presentan en las Figura 2.3 y la Figura 2.4. Se puede observar en ambas figuras que el juego es el mismo y lo que varia es el ambiente en el que se va a llevar a cabo (el ambiente en la Figura 2.4 es una iglesia mientras que en la Figura 2.4 es un parque). Ambos ejemplos muestran como se relaciona cada una de las escenas con lugares propios del ambiente de juego. Esto implica posicionar las preguntas.

Es posible que dos o más escenas transcurran en los mismos lugares (como se visualiza en la Figura 2.4), luego acorde a la situación del alumno se le debe brindar la información correspondiente para que éste continué el juego.

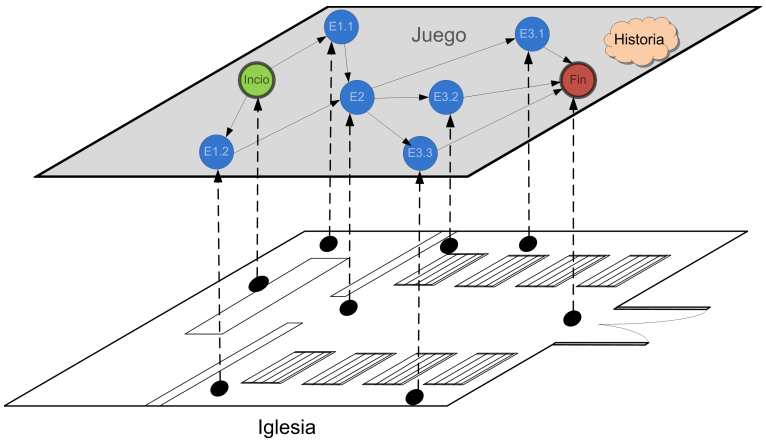

Figura 2.3: Ambiente físico indoor: Iglesia.

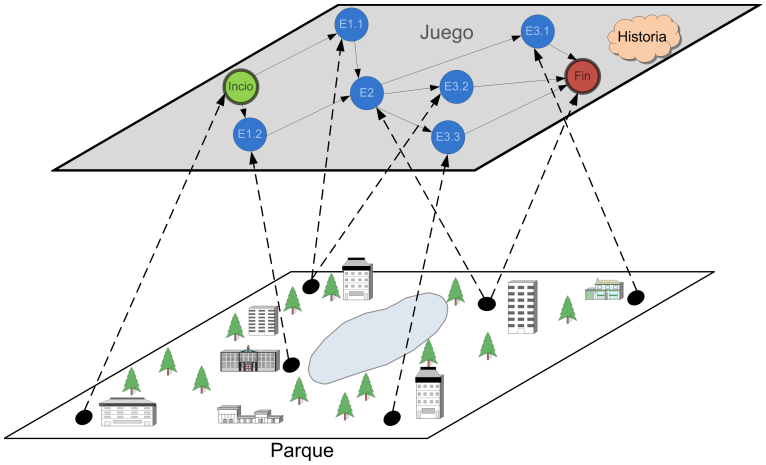

Figura 2.4: Ambiente físico outdoor: Parque.

El ejemplo planteado en las Figuras 2.3 y 2.4 es valido, como se menciona en [1], porque el planteo del juego es independiente del lugar donde se desarrolla el mismo. Si el juego estuviera planteado para realizar trabajo de campo cómo se plantea en [10], el re-uso seria limitado, ya que las preguntas tendrían relación directa con el ambiente de juego.

Acorde a lo descrito anteriormente tenemos dos aspectos claros relacionados con los Juegos Educativos Móviles basados en posicionamiento, por un lado el juego que está compuesto por la historia y las escenas (donde se le plantea a los alumnos alguna actividad, por ejemplo: responder una pregunta), y por otro lado el ambiente físico donde se lleva a cabo el juego. Además, se debe definir la posición donde ocurre cada escena.

2.3. Posicionamiento de los alumnos en el juego

Para este tipo de aplicaciones móviles, como son los Juegos Educativos Móviles basados en posicionamiento, es fundamental conocer la ubicación (o posición) de los alumnos. Para esto, existen diferentes formas de obtener la posición utilizando los dispositivos móviles, por ejemplo: GPS[2], lectura de código 2D, Wifi, entre otros.

Para este trabajo elegimos posicionar al alumno de una manera simple y que sea flexible para ser usado tanto en lugares indoor como outdoor, por esta razón se eligió usar lectura de código de barras 2D[3](de ahora en adelante, lectura de código de barra asumiendo que siempre son de códigos 2D). La Figura 2.5 muestra un ejemplo de la forma que tienen estos códigos.

Figura 2.5: Ejemplo de código de barras 2D.

Para la lectura del código de barra es necesario que el alumno tenga instalado un programa[4] especial el cual utiliza la cámara del dispositivo para ´leer' el código. Acorde a la tipo de código será la acción que tome el programa, por ejemplo, si el código representa una dirección Web (URL), se enviará el requerimiento Web, y como resultado se obtendrá la página Web correspondiente a dicha dirección.

Para posicionar al alumno dentro del juego, primer se deben generar los códigos de barras. Luego, cuando el alumno comienza el juego debe ir leyendo cada uno de los códigos. Supongamos que estos códigos representan direcciones Web, cada vez que un alumno lee uno de estos códigos recibe la página Web con la pregunta correspondiente.

[2]El problema que tiene el GPS es que al entrar en un espacio indoor (por ejemplo: un edificio) y no puede determinar fehacientemente la posición real del alumno. El margen de error puede ser de pocos metros, pero en los juegos este error puede afectar la precisión de las preguntas que se le dan al alumno.

[3]A diferencia de los códigos de barras de una dimensión, es posible poner diferentes tipos de datos dentro del código de barra 2D, no solamente números, por ejemplo: direcciones Web, números de teléfono, códigos alfa numéricos, entre otros.

[4]Existen varios programas para leer código de barras, tales como: QuickMark [19], BeeTagg [2], ScanLife [23], entre otros.

Además, se le puede indicar al alumno mediante un mapa donde esta ubicado. Es decir, la dinámica del juego será la siguiente, cuando el alumno llega a una escena, debe leer el código de barras y una vez leído, se podrá determinar dónde se encuentra y qué información ose le debe otorgar.

La Figura 2.6 muestra la interacción del alumno cuando éste lee un código de barra 2D y la respuesta que recibe del Sistema de Juegos (de ahora en adelante, Sistema), en este caso particular se le muestra al alumno la pregunta asociada al código.

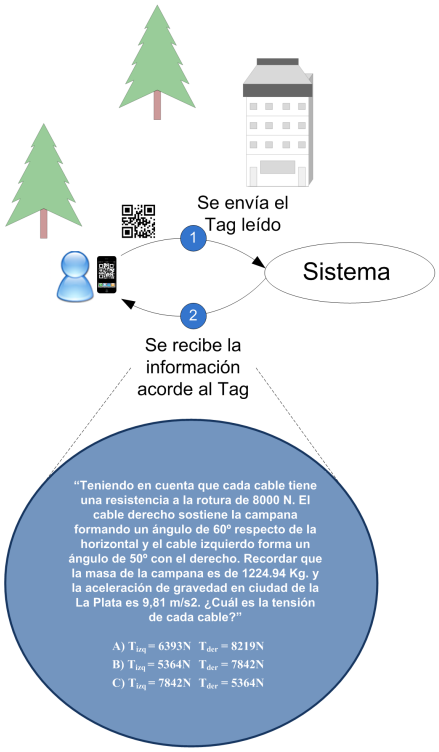

Figura 2.6: Interacción entre el alumno y el Sistema.

2.4. Dinámica de los juegos

Los Juegos Educativos Móviles basados en posicionamiento, requieren que el alumno se vaya moviendo en el ambiente físico del juego, y a medida que llega a cada escena debe leer un código de barra, para poder de esta manera interactuar con el juego.

Supongamos que un alumno juega el juego presentado en la Figura 2.4, es

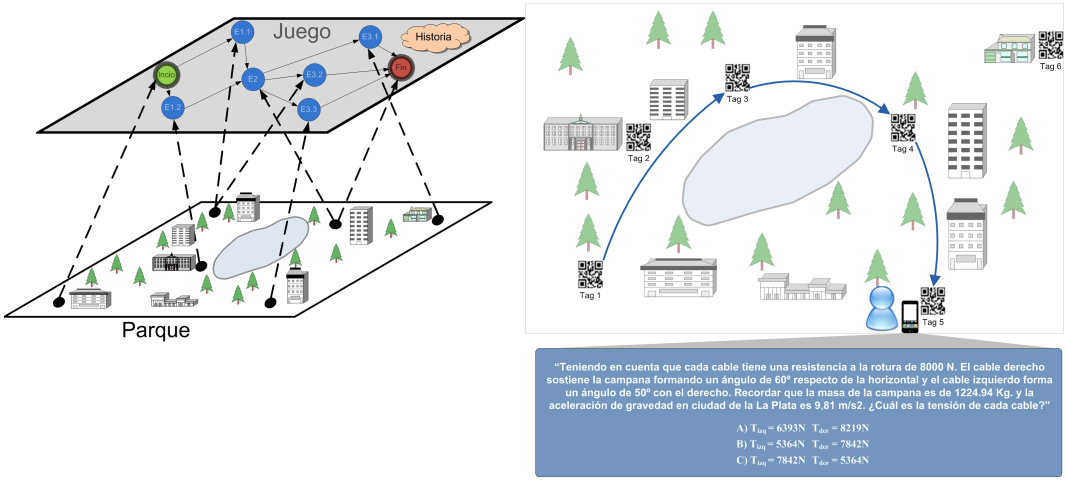

Figura 2.7: Recorrido de un alumno.

decir que tiene una determinada historia con sus preguntas y el ambiente físico es un bosque. Primero, el alumno debe leer el código inicial (del juego), acorde a este código el Sistema le dará al alumno información referente al juego, como por ejemplo, una descripción del mismo. Luego, el Sistema le mostrará el abanico de posibles escenas disponibles, supongamos que el alumno elige ir a la escena *E1.1*. Acorde a esta elección del alumno, el Sistema le indica mediante un mapa el camino que debe realizar para llegar a su destino.

Una vez que el alumno llega al lugar (correspondiente a la escena *E1.1*), utiliza su dispositivo móvil para leer el código. Al hacerlo, el Sistema le retorna la pregunta correspondiente a la escena *E1.1*. Una vez que responde, el Sistema le indica con un mapa hacia dónde debe dirigirse para ir a la próxima escena, en este caso *E2*.

Luego de unos instantes de caminar, el alumno llega al lugar de la escena *E2* y completa satisfactoriamente la tarea propuesta. Luego debe optar hacia qué escena desea dirigirse (*E3.1, E3.2* o *E3.3*). Supongamos que el alumno decide ir a la escena *E3.3*, recibe un mapa con el camino al destino y luego llega al destino. La Figura 2.7 ilustra el recorrido del alumno hasta el momento, además se puede observar la pregunta que recibe el alumno cuando leer el código de barra asociado a la escena *E3.3*. Una vez que el alumno responde la pregunta el Sistema le indica cómo llegar a la última escena en el juego.

Supongamos que el alumno en vez de llegar a la escena final (por algún motivo, por ejemplo: porque se perdió) llega al lugar de la escena *E3.1* y lee el código correspondiente a dicha escena. Esta situación se puede observar en la Figura 2.8. Se puede apreciar que el alumno recibe un nuevo mapa, que le indica a donde

debe dirigirse para continuar el juego (en este caso, la escena final).

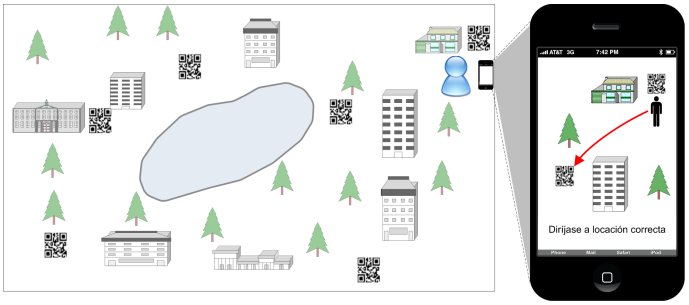

Figura 2.8: El participante llega a una escena que no le corresponde.

Cuando el alumno lea el código de la escena final, el Sistema reconocerá que es el final del juego y le indica su desempeño dentro del mismo. Es decir, el alumno recibirá información de cómo contestó cada una de las preguntas. Para esto, el Sistema deberá guardarse por cada alumno cómo fueron sus respuestas a lo largo del juego. Además, el Sistema también puede guardar las actividades del los alumnos para poder realizar luego un análisis más detallado del comportamiento de los mismos.

Cabe aclarar que los diferentes lugares donde se lleve a cabo el juego y la secuencia de escenas que los alumnos elijan para un juego, son los elementos que podrán variar cada vez que se juegue el juego. Es decir, un alumno puede jugar el mismo juego más de una vez, y variar sus elecciones, esto hace que la experiencia del alumno sea diferente.

2.5. Análisis del comportamiento de los alumnos

En esta sección se analizaran algunos aspectos relacionados al ejemplo presentado en las secciones anteriores desde la perspectiva del comportamiento del alumno. Se pudo observar, en el ejemplo presentado, que cuando un alumno participa en este tipo de juegos básicamente tiene dos actividades bien diferenciadas:

- Leer un código de barra e interactuar con la información que le brinda el Sistema.
- Trasladarse físicamente (camina) desde una escena a otra.

Las dos actividades mencionadas son notablemente distintas y caracterizan momentos diferentes durante el juego. Es posible distinguir, que dependiendo de

cada una de ellas, el Sistema le entregará información diferente al alumno. Si hacemos un análisis sobre la primera de las acciones *(Leer un código e interactuar con la información que le brinda el Sistema)* podemos determinar que se puede detallar la misma con más nivel de detalle, como se describe a continuación:

- **El alumno lee el código correcto (que le correspondía)**. Esto involucra:
  1. El Sistema le envía al alumno una pregunta.
  2. El alumno contesta la pregunta.
  3. Si hay más de una escena siguiente,
     a) El Sistema le envía al alumno una lista de posibles escenas siguientes, para que éste elija a dónde desea dirigirse.
     b) El alumno indica la escena (de la lista) a dónde desea dirigirse.
  4. El Sistema le brinda al alumno un mapa para que se dirija a la próxima escena.
- **El alumno lee un código incorrecto.** Esto involucra que el Sistema le brinda al alumno un mapa para que se dirija a la escena correcta.

Las actividades que realiza el alumno también están determinando el estado actual del mismo, por ejemplo, respondiendo una pregunta, caminando, entre otros. Acorde al estado del alumno en un momento dado, el Sistema puede determinar qué acciones puede realizar el alumno. Por ejemplo, mientras se encuentra caminando hacia una escena, el Sistema no le brindará ninguna pregunta o información.

El Sistema puede determinar donde está cada alumno y que actividad está realizando en un momento dado. Esta información se va obteniendo acorde a la interacción que realiza el alumno con el Sistema, mediante por ejemplo, leer un código o contestar una pregunta. Como a través del Sistema pasa toda esta información, sería deseable que el Sistema guarde el historial del alumno para poder saber cuál fue el comportamiento del mismo. A su vez, esto nos permitiría ver a las respuestas de todas las preguntas, y saber cómo contestó.

2.6. Múltiples alumnos en un juego

Tomando el ejemplo presentado en la Sección 2.5, veamos ahora la dinámica del mismo cuando hay tres alumnos jugando. Con este escenario podremos apre-

ciar que el Sistema deberá distinguir la situación (estado actual) de cada uno de los alumnos para brindarle la información adecuada a cada uno. Para esto, el Sistema deberá almacenar el comportamiento de cada uno de los alumnos.

En la Figura 2.9 se puede apreciar tres alumnos que ya han leído el código del inicio del juego, y cada uno ha elegido la escena a dónde va a dirigirse. Se puede observar que el alumno A decide ir a la escena *E1.1* mientras que los alumnos B y C deciden ir a la escena *E1.2*. Cada alumno, recibe un mapa con el camino para llegar a la escena elegida.

Hasta este punto del juego, se les ha proporcionado a los alumnos parte de la historia. El desarrollo de la misma irá avanzando en cada una de las escenas a dónde decidan dirigirse. Puede pasar que los alumnos coincidan o no en las escenas que seleccionan para continuar el juego, como es el caso de los alumnos B y C en la 2.9.

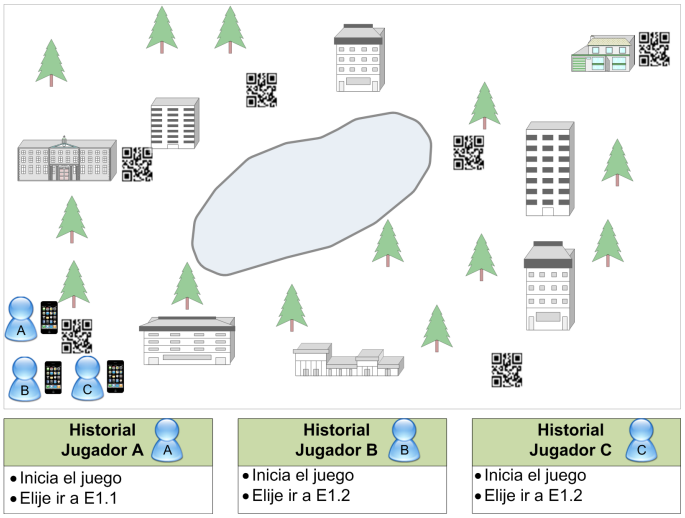

Figura 2.9: Todos los alumnos están en el inicio del juego.

Supongamos que cada alumno llega a su respectiva escena, el Sistema les otorga a cada uno la pregunta existente en la escena correspondiente. Los alumnos deben responder la pregunta. Luego, el Sistema les indica mediante un mapa que deben dirigirse a la escena *E2*, esta situación queda reflejada en la Figura 2.10. Podemos ver que el Sistema reconoce que existen jugadores que se encuentran en una misma escena y que hay otro que no. El Sistema es capaz de indicar para cada uno de los alumnos el mapa correspondiente para que los jugadores se puedan movilizar sin problemas.

El juego continua, y sólo llegan a la escena *E2* los alumnos A y B como puede apreciarse en la Figura 2.11. Del alumno C lo único que se conoce es que debía

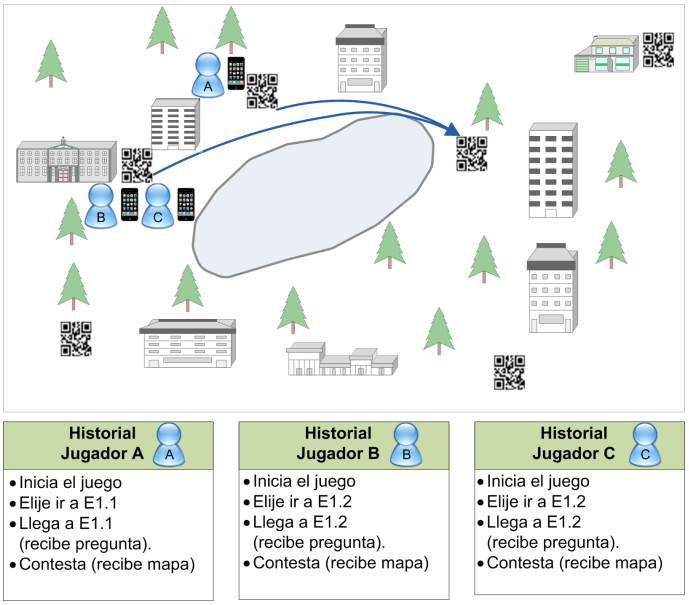

Figura 2.10: Todos los alumnos están en la primera escena elegida.

dirigirse a la escena *E2* pero no se puede determinar dónde está actualmente ubicado, ya que el ultimo registro es que estaba en la escena *E.1.2*. Es decir, no es necesario que todos los alumnos avancen y lleguen a las escenas al mismo tiempo, el Sistema debe mantener el estado actual de cada uno de los alumnos para proporcionarles la información adecuada.

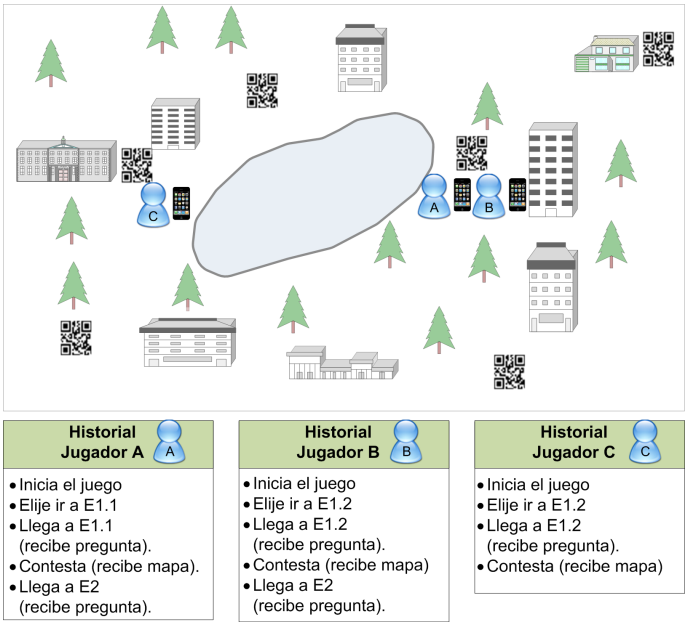

Figura 2.11: Dos jugadores llegan a la misma escena.

Luego, tanto el alumno A como B contestan la pregunta presentada en la escena *E2*. Ambos alumnos eligen dirigirse a la escena *E3.3*, el Sistema les brinda un mapa para que lleguen a esta escena, y sólo el alumno A llega a dicha escena. Cuando el alumno A llega a la escena *E3.3* recibe la pregunta correspondiente como se puede apreciar en la Figura 2.12. Se puede observar que el alumno B se encuentra caminado pero no se puede determinar con precisión en que parte del camino está hasta que no lea un código.

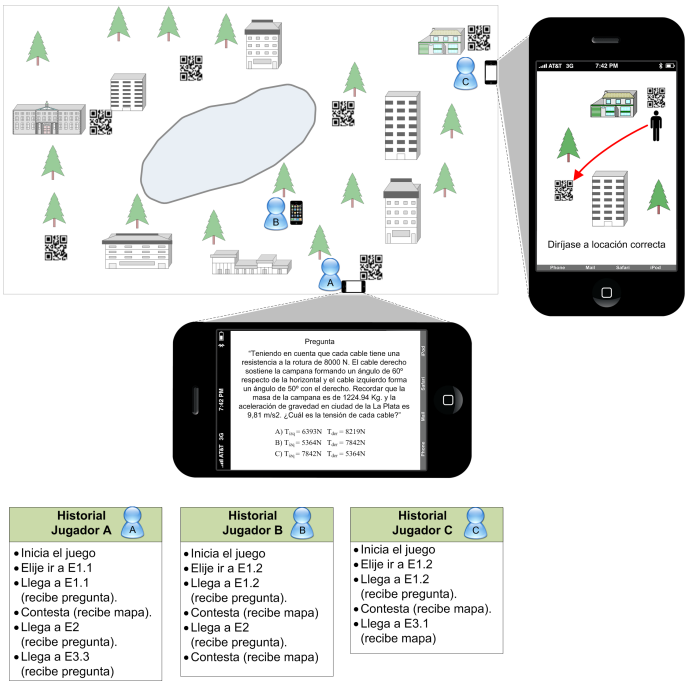

Figura 2.12: El Jugador C llega a una escena incorrecta y el Jugador A interactua con el sistema.

En cuanto al alumno C, lee el código correspondiente a la escena *E3.1* (código incorrecto), al no ser la escena a la cual le correspondía ir, el Sistema le brinda un mapa para que se dirija a la escena correspondiente como se puede apreciar en la Figura 2.12.

A partir de las situaciones (de juego) antes descriptas se puede tener un panorama general de cómo funcionan este tipo de juegos. El hecho que un alumno haya terminado el juego no implica que el resto deba dejar de jugar, salvo que por construcción del juego así se lo desee.

Se pudo apreciar que el juego y el ambiente físico fueron iguales para todos los alumnos y cada uno fue eligiendo las escenas que quería recorrer y así armo su propia vivencia del juego. Por lo tanto, podemos distinguir tres elementos o conceptos distintos, el juego, el ambiente físico y los alumnos. El juego mantiene

su estructura (historia, escenas, relación entre las escenas, entre otros.) como algo estático, es decir algo que es inalterable. El ambiente físico, es el lugar dónde se realiza el juego y es dónde estarán posicionados los códigos de barras. Cada código tendrá asociado que escena se lleva a cabo en ese lugar. En cuando a los alumnos, estos se comportan de manera diferente, de una manera más proactiva y dinámica. Cada alumno es distinto, dado que puede desenvolverse de manera completamente diferente uno de otro. Es decir, que puede responder de manera distinta las preguntas o decidir ir hacia otras escenas y con ello alterar el resultado final como se vio en el juego presentado en esta sección.

Hemos visto a lo largo de este capítulo las características de los Juegos Educativos Móviles basados en posicionamiento. Además, se ha mostrado cómo es el comportamiento de los alumnos en este tipo de juegos, como así también la interacción que deben realizar. Se destaco la importancia de tener el estado actual de cada alumno como así también el historial de actividades que fue realizando.

Capítulo 3

Modelado del Usuario

En este capítulo se presentará un Modelo del Usuario para Juegos Educativos Móviles basados en posicionamiento. Para lograr esto, primero mostraremos en la Sección 3.1 un modelo general de los juegos y luego en la Sección Modelado-DelUsuario se presentará el Modelo del Usuario en cuestión.

3.1. Modelo General de los Juegos Educativos Móviles

Si bien el foco de este capítulo será presentar un Modelo del Usuario, para lograr entender mejor cómo este modelo es parte de los juegos se representarán algunos conceptos básicos de este tipo de juegos, por ejemplo: el juego, las escenas, los ambientes de juego por nombrar algunos.

En la Figura 3.1 se presentan los conceptos básicos que definimos para representar los juegos acorde a lo descrito en el Capítulo 2. Se puede apreciar, en esta figura, la clase *Player* la cuál representa a los jugadores. La clase *Player* será la encargada de mantener la información del jugador como así también la actividad del mismo a lo largo del juego. Se puede apreciar que esta clase tiene definido un nombre y un puntaje[1]. En la Sección 3.2 mostraremos cómo esta clase es parte

[1]Por simplicidad, este modelo suma un punto cuando el jugador responde una pregunta correcta.

de nuestro Modelo del Usuario, y qué otra información tiene relacionada.

Cabe aclarar que en este trabajo un jugador será un alumno, pero al definirse de manera general este modelo puede ser usado para otro tipo de juego que no sea educativo. En esté capítulo se mencionará el concepto de jugador teniendo en cuenta que nos estamos refiriendo a un alumno.

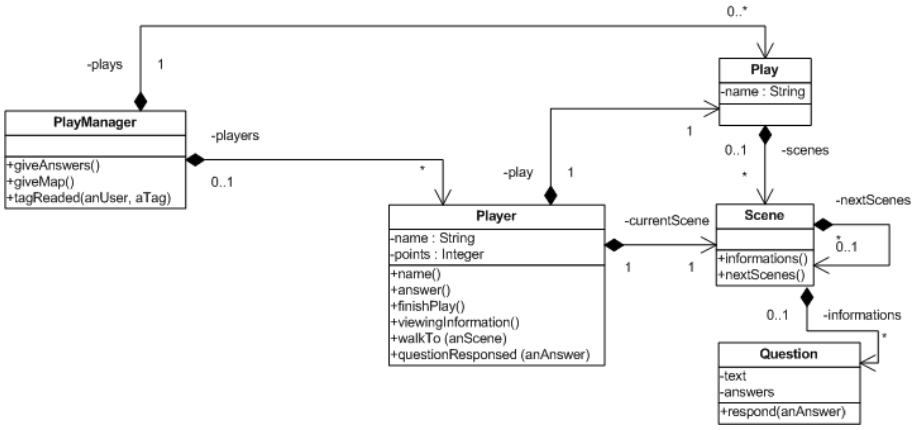

Figura 3.1: Modelo de los Juegos.

Respecto a la Figura 3.1, podemos mencionar que los juegos son representados por la clase *Play*. Ya hemos mencionado en el Capítulo 2 que cada juego cuenta con una división explícita de la historia, esta división son las escenas. Las escenas son modeladas con la clase *Scene*. Cada juego conoce todas sus escenas y cada escena conoce sus próximas escenas. De esta manera queda modelada la estructura del juego. Cabe mencionar que dentro de cada escena tendremos las informaciones o preguntas que serán brindadas a los jugadores. Por simplicidad, sólo se modelaron las preguntas con la clase *Question*. Podemos observar en la Figura 3.1, que los jugadores sólo pueden jugar un juego por vez, por esta razón es necesario saber en qué juego se encuentra jugando (relación play). Además, cada jugador conoce la escena actual en la que se encuentra (relación currentScene) y la siguiente escena a dónde debe dirigirse (relación nextScene) si es que ya fue indicada[2]. La Figura 3.1, define, además, la clase *PlayManager* para agrupar toda la información de los juegos y los jugadores. Esto permite que haya más de un juego creado y jugadores jugando en distintos juegos.

Para una mejor comprensión de los conceptos presentados, veamos cómo instanciar el modelo presentado en la Figura 3.1. La Figura 3.2 muestra un instanciación realizada acorde al ejemplo que se presentó en la Figura 3.1 (Capítulo 2). Se puede apreciar que hay un juego, este juego contiene escenas (cada una

[2]Puede pasar que cuando un jugador está, por ejemplo, contestando una pregunta todavía no tiene una escena siguiente a la que deba dirigirse.

define sus posibles escenas siguientes). En particular, la escena *E.3.3* muestra la pregunta que tiene asociada (por simplicidad el resto de las escenas no muestran las preguntas que tienen asociadas). Además, se puede apreciar que el jugador tiene su escena actual (escena *E.3.3*).

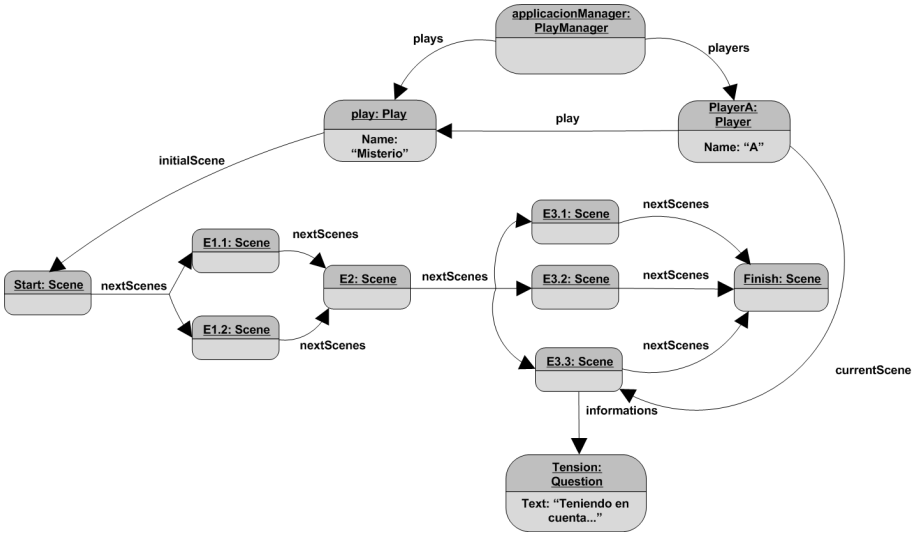

Figura 3.2: Diagrama de instancias reducido del juego presentado en el Capítulo 2.

Como se mencionó en el Capítulo 2, el ambiente físico (o ambiente de juego) es fundamental en este tipo de juegos. En la Figura 3.3 se puede apreciar cómo este concepto fue modelado con la clase *PlayEnviroment*. Esta clase conoce a qué juego pertenece y contiene la representación del lugar físico dónde se lleva a cabo el juego. Se puede apreciar en dicha figura que la clase *Player* conoce tanto a la clase *Play* como *PlayEnviroment*, de esta manera se puede determinar que juego está jugando y en que ambiente físico se está jugando el mismo. Esto es necesario ya que un mismo juego puede llevarse a cabo en distintos ambientes físicos.

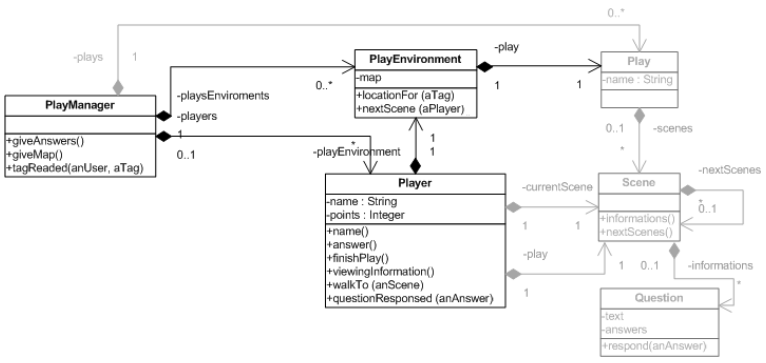

Figura 3.3: Ambiente de Juego como un aspecto relevante en los Juegos Educativos Móviles basados en posicionamiento.

La clase *PlayEnviroment* tiene la responsabilidad de determinar, por ejemplo:

- Dónde está ubicado, un código de barra. Esto permite determinar, cuándo un jugador lee un código dónde se encuentra ubicado dentro del ambiente de juego. Esto se puede realizar porque cada código tiene asociado una escena, y cada escena tiene una ubicación dentro del ambiente físico.
- Dónde está ubicada una escena particular. Cada escena se ubica en un lugar determinado dentro del ambiente físico. Esto permite, determinar dónde está ubicada la próxima escena del jugador.
- El mapa que se le debe brindar al jugador para ir de una escena a otra.

3.2. Modelo del Usuario en los Juegos Educativos Móviles

Veamos en esta sección cómo definir el Modelo del Usuario tomando como punto de partida la clase *Player* definida en la Figura 3.1. Recordemos que esta clase conoce su escena actual y su siguiente. Usando esta información la clase *PlayManager* puede determinar dónde está el jugador dentro del ambiente físico del juego, ya que una escena está ubicada dentro de un *PlayEnviroment*. Además, la siguiente escena sirve para saber a dónde debe dirigirse un jugador (una vez que cumple la actividad que tenia pautada en la escena actual).

El jugador puede ir pasando por diferentes estados, como ya se mencionó en el Capítulo 2, por ejemplo, puede estar caminando, leyendo un código de barra, contestando una pregunta, por nombrar algunos. Para representar los estados de los jugadores se definen los mismos como se muestra en la Figura 3.4[3]. La clase *PlayManager* reaccionará acorde al estado actual que tenga el jugador, para brindarle así la información correspondiente.

La clase *PlayerState* respeta el patrón de diseño que lleva el mismo nombre (State [9]). En particular, se modelaron los estados:

- *LookingATag*: representa que el jugador leyó un código de barra.
  - *Awsering*: representa que el jugador leyó un código que tiene asociada una pregunta.
  - *ViewingInformation*: representa que el jugador leyó un código que tiene asociada una información.

[3]Por simplicidad, las relaciones currentScene y nextScene se especificaron como atributos de la clase *Player*.

- *Walking*: representa que el jugador está caminando.
- *Finishing*: representa que el jugador finalizó el juego.

Esta jerarquía de estados se puede extender en el caso de necesitar representar otros estados o estados más específicos. Es decir, se definieron los estados de una manera flexible para que pueda evolucionar el modelo. Cada estado tendrá sus datos propios y brindarán el comportamiento correspondiente al jugador.

Mientras se realiza un juego, el jugador va a ir cambiando de estados acorde a cómo se comporte en el juego. La Figura 3.5 muestra un diagrama de transición de estados, especificando cuáles son las transiciones posibles por las que puede ir pasando el jugador. Se puede observar que tenemos un estado inicial (*Start*) y uno final (*Finishing*). Vemos que el estado *Walking* no puede ser usado como estado inicial, esto significa que primero se debe leer al menos un código de barras 2D para comenzar el juego en cuestión y luego comenzar a jugarlo.

A continuación se describe qué involucra cada una de las transiciones especificadas en la Figura 3.5.

A Terminada la inicialización del jugador se procede a brindarle un mapa para que se dirija a la primer escena.

B Cada vez que, el jugador, lee un código de barras 2D se pasa a este estado.

C El jugador leyó un código de barras 2D, que era el que le correspondía ver, y el mismo tiene asociada una pregunta, la cual se le muestra en su dispositivo.

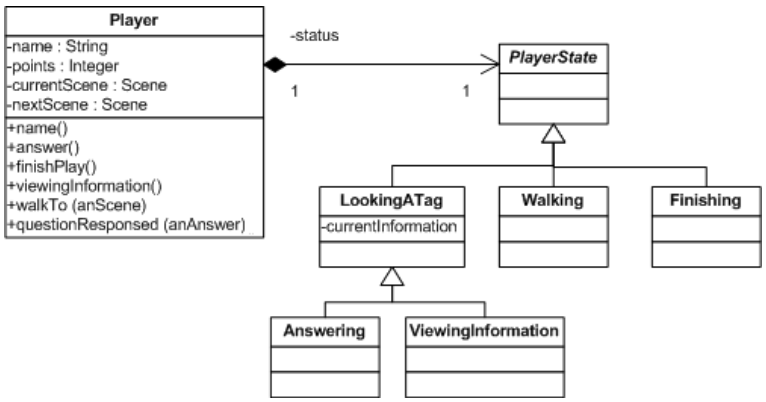

Figura 3.4: Estados del Jugador.

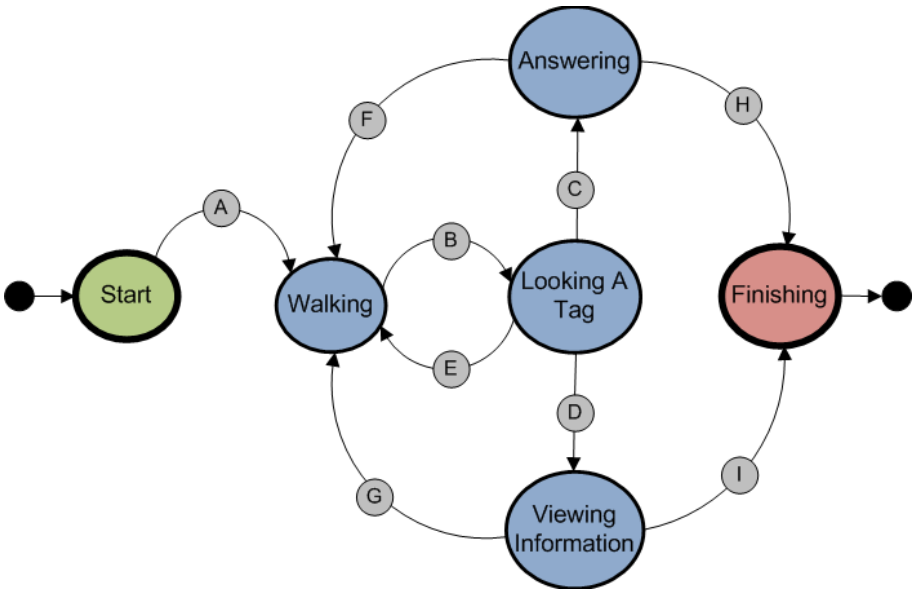

Figura 3.5: Estados de los jugadores.

D El jugador leyó un código de barras 2D, que era el que le correspondía ver y el mismo tiene asociada una información, la cual se le muestra en su dispositivo.

E El jugador leyó un código de barras 2D, que no era el que le correspondía, se le informa de ello y se le brinda un mapa para guiarlo al destino correspondiente.

F Luego de contestar una pregunta debe dirigirse a otra ubicación, para ello se le presenta un mapa para guiarlo.

G Luego de ver una información el jugador debe dirigirse a otra escena, para ello se lo guía a través de un mapa.

H Una vez que contesta una pregunta, y está era la última del juego.

I Una vez vista una información, siendo ésta la última del juego.

Como se mencionó en el Capítulo 2, es importante tener almacenado el historial de actividades que fue realizando el jugador, para poder determinar, por ejemplo, cómo respondió cada pregunta. Por esta razón se creó la clase *History*, la cual mantiene todas las actividades que realiza el jugador a lo largo del juego. Se puede apreciar en la Figura 3.6 cómo se modeló este concepto de manera desacoplada del jugador. Las actividades que realiza el jugador se van registrando como eventos, para esto se creó una jerarquía de eventos (*Event*). Cada evento representa una actividad particular que realizó el jugador y este guarda la información

correspondiente a la actividad realizada. Se puede apreciar en la Figura 3.6 los eventos mínimos que se definieron para los juegos que son de nuestro interés.

Los eventos presentados en la Figura 3.6 reflejan las actividades que realizó un jugador. Veamos en detalle que representa cada uno de los eventos modelados:

- StartPlay: Una vez que se iniciaron los componentes necesarios para que el jugador pueda comenzar a jugar, queda registrado en qué juego comenzó a jugar (atributo play)
- Arrive: Representa la llegada de un jugador a una escena (que puede ser a la que debía llegar o no), quedando asentado cuál fue dicha escena (atributo scene).
- ViewingQuestion: El jugador se encuentra viendo una pregunta quedando almacenada la misma en el atributo question.
- ViewingInformation: El jugador se encuentra viendo información la cual queda registrada en el atributo information.
- QuestionAnswered: El jugador ha contestado una pregunta. Queda almacenada la información con respecto a la pregunta (atributo question) y la respuesta (atributo answer) que realizó el jugador junto con el resultado de la misma (atributo result, el resultado se calcula dinámicamente dado que el tipo de preguntas es múltiple choice).
- AddPoints: El jugador contestó satisfactoriamente una pregunta y se lo premia otorgándole una cantidad de puntos la cual queda registrada en el atributo pointsToAdd.
- Redirection: El jugador ha llegado a una escena que no le corresponde (atributo arrivedScene) y se registra el redireccionamiento hacia la escena correcta (atributo correctScene).
- NextScene: Se ha modificado la próxima escena a la cual debe dirigirse el jugador (atributo scene).
- FinishPlay: Generado en el momento en que un jugador termina el juego, reflejando que un jugador a finalizado el juego.

Para una mejor comprensión de los eventos descriptos, ejemplifiquemos cómo se van usar en un juego. Supongamos que se tiene un jugador que se encuentra contestando una pregunta y lo hace de manera correcta, el Sistema le asignará un punto más al jugador. Para representar esta información se generan dos eventos

Figura 3.6: Historial del Jugador.

uno al contestar la pregunta y otro asociado a la modificación del puntaje, estos eventos serán almacenados por el History. Cada uno de los eventos presentados en la Figura 3.6 se generan en relación a algún cambio de estado del jugador. Se puede generar más de un evento como consecuencia de un cambio de estado, como se ejemplifico anteriormente. Veamos en la Tabla 3.1 que eventos se deben generar (y almacenar en el historial del jugador) para cada una de las transiciones especificadas en la Figura 3.5 y que acciones conllevan cada uno de ellos.

Anteriormente, en la Sección 3.1, se mencionó que la clase *PlayManager* es la encargada de determinar que información brindarle al jugador acorde al código de barras 2D que éste leyó. Esto lo realiza delegando está responsabilidad a la clase *PlayEnviroment*. Pero no siempre que el jugador lee un código de barras 2D debe recibir, por ejemplo, la pregunta asociada al mismo. La pregunta sólo la debe recibir si esa escena (que se corresponde con el código de barras que leyó) es a la que le correspondía ir, en caso contrario se le debe indicar dónde está ubicada la escena a la que debía dirigirse. La clase *PlayManager* debe determinar si cuando un jugador lee un código de barra, hay que brindarle la pregunta asociada o indicarle a dónde está la escena correcta. Para esto, la clase *PlayManager* usa el estado actual del jugador (por ejemplo, *Walking*) y la escena a la que se debería dirigirse para chequear si el código de barras 2D leído se corresponde o no con esta escena. Además, la clase *PlayManager* es la encargada de actualizar el historial de actividades del jugador (agregando los eventos correspondientes), como así también cambiar el estado del mismo acorde a las acciones que realiza el jugador.

En la Figura 3.7 se puede apreciar el Modelo del Usuario que se fue creando

Transición	Eventos	Acciones
A	*StartPlay*	Cambia el estado al jugador a *Walking* y se le establece la primera escena a la cual debe dirigirse.
B	*Arrive*	Cambia el estado a *LookingATag* y se establece la escena actual.
C	*ViewingQuestion*	Cambia el estado a *Answering.*
D	*ViewingInformation*	Cambia el estado a *ViewginInformation.*
E	*Redirection*	Cambia el estado a *Walking.*
F	*QuestionAnswered* si contestó satisfactoriamente *AddPoints*	Cambia el estado a *Walking* y se le establece al jugador la siguiente escena a la cual debe dirigirse.
G	*NextScene*	Se le establece al jugador la siguiente escena a la cual debe dirigirse.
H	*QuestionAnswered*, si contestó satisfactoriamente *AddPoints* y además *FinishPlay*	Cambia el estado a *Finishing.*
I	*FinishPlay*	Cambia el estado a *Finishing.*

Cuadro 3.1: Eventos y acciones realizadas en cada transacción.

en forma incremental, destacando las clases relevantes de este modelo. Cabe aclarar que las clases del Modelo General del Juego presentado en la Sección 3.1 se muestran en esta figura para lograr integrar mejor los conceptos del Modelo de Usuario.

Teniendo en cuenta el ejemplo presentado en la Figura 2.9 del Capítulo 2 veamos cómo usando esa información se puede instanciar el modelo presentado (Figura 3.7). Para esto, en la Figura 3.8 se muestra el modelo de instancias. Se puede apreciar que hay un juego, el juego se desarrolla en un ambiente en particular. En este ambiente de juego hay tres jugadores jugando el mismo, todos en estado *Walking*, además cada uno de los jugadores tiene su historial de eventos.

Usando la instanciación de la 2.9 veremos mediante un diagrama de secuencia en la Figura 3.9 cómo reacciona el Sistema cuando un jugador lee un código de barras 2D. En el diagrama se puede observar que se analizan los dos casos posibles: que el código leído sea el correcto o que no lo sea. Para los ejemplos utilizaremos primero al Jugador A el cual llega correctamente a la escena que le corresponde, y luego al Jugador C el cual llega a una escena incorrecta.

Analicemos la Figura 3.9, tanto para el Jugador A como para el Jugador C se sucederán estos primeros pasos que se describen a continuación. La secuencia de mensajes del 1.1 al 1.3 se mantiene igual dado que siempre se deben realizar las mismas acciones ante todos los jugadores al momento de leer un código de barras 2D. El Sistema debe obtener la instancia de la escena correspondiente al código leído por el Jugador (cualquiera sea este) mediante el mensaje getSceneFor(). Una vez obtenida la escena en la cual se encuentra el jugador (currentScene), el Sistema debe actualizar la posición al jugador y actualizar el estado en el que se encuentra. Para ello le envía el mensaje arrive() junto con la nueva posición. Dentro de este

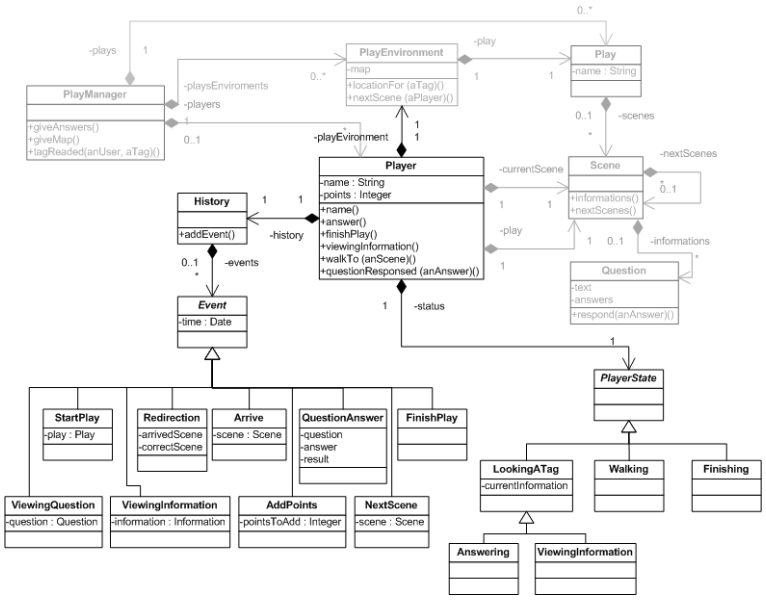

Figura 3.7: Modelo del Usuario.

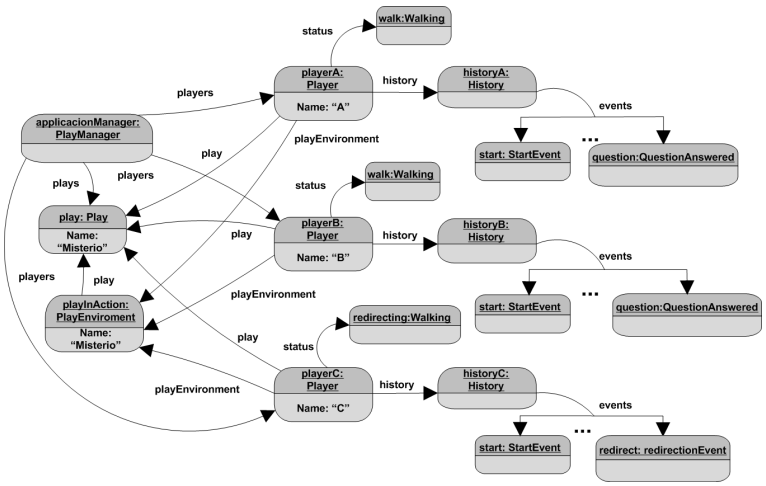

Figura 3.8: Ejemplo de instanciación del modelo propuesto.

mensaje se cambia el estado del jugador por *LookingATag* hasta que se pueda saber si ha llegado o no a la escena correcta. Ahora el Sistema debe averiguar si el jugador llegó a la escena que le corresponde. Para ello, obtiene la escena a la cual se debe dirigir enviándole el mensaje getNextScene() almacenando el resultado en playerNextScene. Ahora pueden ocurrir dos casos:

- El jugador llegó a la escena correcta. En nuestro caso el Jugador A. En este caso las escenas currentScene y playerNextScene son las mismas con lo cual se debe procesar la información que hay en la escena. Para esto, se debe pedir la información a la escena y luego dependerá de qué tipo de información es para que se cambie al estado correspondiente al jugador. El Sistema envía el mensaje discoverInformation() y este delegará en la información para resolver este problema. Para nuestro ejemplo, como el jugador ha llegado a la escena que le corresponde (para nuestro ejemplo la escena *E2.2*) y en ella hay una pregunta se le cambiará el estado por *ViewingQuestion* y se le brinda la pregunta.
- El jugador llegó a una escena incorrecta. Como es el caso del Jugador C. En este caso el Sistema cuando verifica si las escenas son las mismas dará falso con lo cual debe redireccionar al jugador hacia la escena correcta. Debe primero cambiar el estado del jugador por *Walking* para referenciar que debe trasladarse físicamente hacia otro lugar. Luego debe calcular y generar el mapa correspondiente para ayudar al jugador. Este mapa es entregado al jugador.

La Figura 3.10 muestra cómo quedan las instancias luego de las actividades relacionadas al Jugador A y al Jugador C. Vemos como quedan reflejados los cambios en los jugadores, donde el historial se ha modificado (agregándose nuevos eventos) al igual que el estado que ha pasado por diferentes tipos. En el caso del Jugador A se ha agregado un evento *ArriveEvent* y se ha modificado el estado por *ViewingQuestion*. En el caso del Jugador C a su historial de eventos se ha agregado el evento *Redirection* y su estado es de *Walking*. Cabe aclarar que el estado por más que es el mismo, al leer el código de barras de 2D se ido modificando.

Continuando con el ejemplo, la Figura 3.11 muestra la secuencia que se sigue cuándo el Jugador A contesta la pregunta que le fue previamente entregada. El jugador envía la respuesta al Sistema y este primero debe modificarle el estado al jugador por *Answering*. Luego debe verificar si la respuesta fue correcta o no, para ello el Sistema envía el mensaje checkAnswer() y almacena el resultado en la variable local al mensaje isCorrect(). En caso que sea correcta (como en nuestro ejemplo) se le otorgarán los puntos correspondientes al jugador con el mensaje addPoints(). El Sistema determina cuál es la próxima escena a la cual debe dirigirse

y se la asigna al jugador a través del mensaje computeNextScene() y setNextScene() respectivamente. El Sistema le debe cambiar el estado al Jugador A (quien realizó

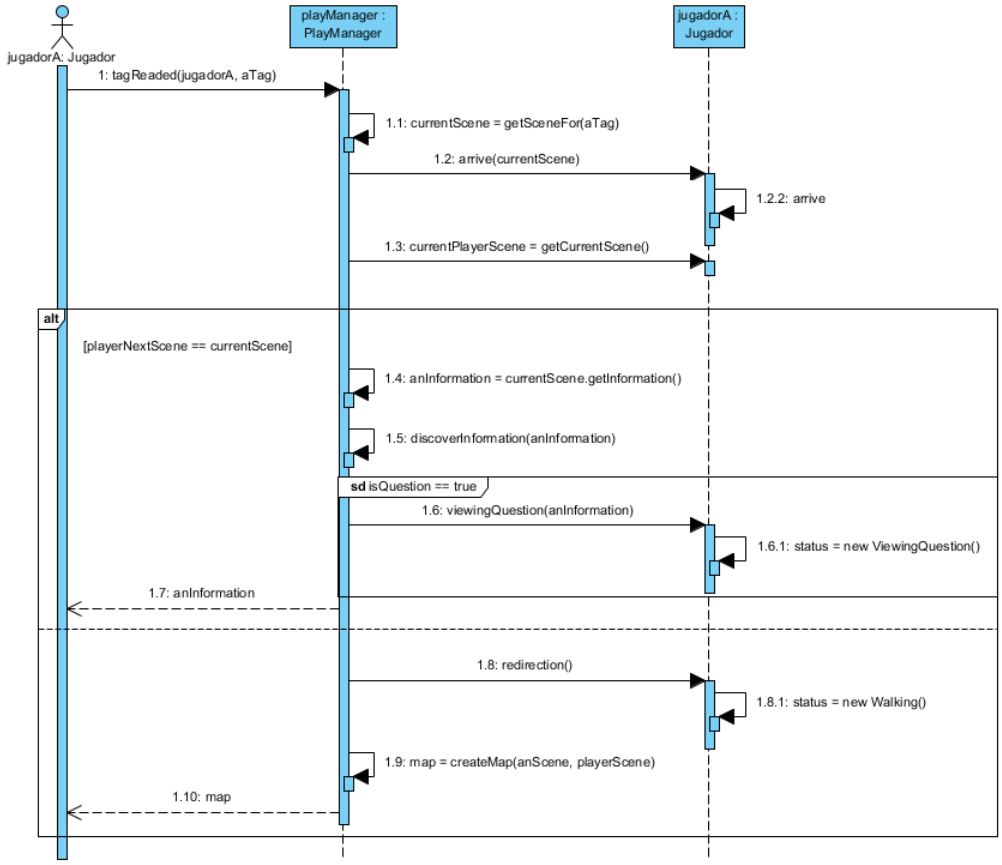

Figura 3.9: Un jugador lee un código de barras.

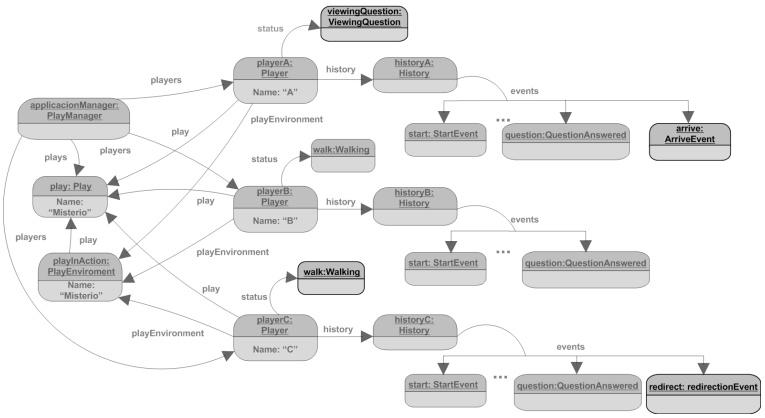

Figura 3.10: Diagrama de instancia del modelo propuesto luego de las actividades de los jugadores.

la respuesta) por *Walking*. Por último, el Sistema le debe calcular y generar un mapa al jugador, el cual deberá seguir para llegar a la próxima escena (en este caso la *E3.3*).

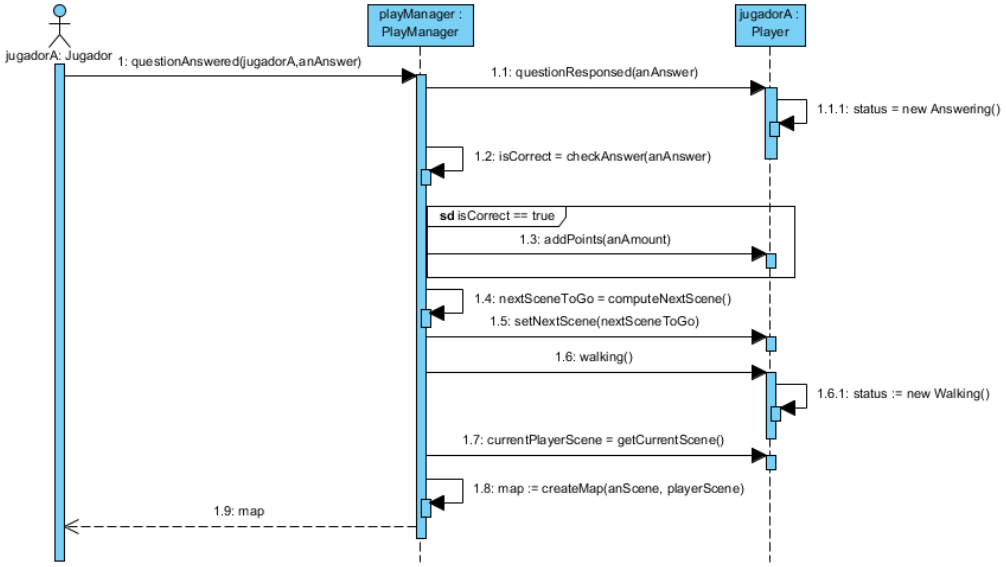

Figura 3.11: Un jugador contesta una pregunta.

De esta manera, queda presentado un Modelo del Usuario para ser utilizado en los Juegos Educativos Móviles basados en posicionamiento que son el foco de este trabajo. Cabe destacar que este modelo define el estado actual del jugador como así también el historial de actividades que el mismo fue realizando. En particular, del jugador se conoce que juego está jugando y en que ambiente físico lo esta jugando, como así también la escena actual y siguiente (si es que ya se le indicó a dónde debe dirigirse).

Capítulo 4

Problemáticas en Juegos Educativos Móviles Basados en Posicionamiento

En este capítulo se describirán las problemáticas involucradas a los Juegos Educativos Móviles basados en posicionamiento. Estas problemáticas fueron determinadas a partir de pruebas prototípicas que permitieron detectar algunas necesidades asociadas a este tipo de juego. Se mencionará en la Sección 4.1 las características básicas del prototipo de juego usado para detectar las problemáticas que se enuncian posteriormente en la Sección 4.2.

Cabe recordar, que el foco de este trabajo esta puesto en brindar al docente las herramientas necesarias para que este tipo de juego se ponga en práctica, tratando de que los eventuales problemas que surjan durante el juego puedan ser resueltos, y lograr así que los alumnos no se frustren en el uso de los mismos. En la Sección 4.2 se propone una posible solución para que el docente pueda contar con una herramienta que le brinde soporte a las problemáticas detectadas. En particular, la herramienta permitía hacer monitoreo de los alumnos, como así también recuperación de fallas surgidas durante el transcurso del juego.

4.1. Juego Educativo Móvil usado cómo prototipo para detectar problemáticas

En esta sección se especifican las características del prototipo de juego usado para detectar las problemáticas involucradas en los Juegos Educativos Móviles basados en posicionamiento. Para el prototipo se uso como lenguaje de programación Smalltalk [6], y en particular, el ambiente de programación VisualWorks [7]. El prototipo se desarrollo como una aplicación Web, la cual se definió el Framework Web Seaside [24], el cual es compatible con VisualWorks.

El prototipo (de juego) se definió en base al patrón de diseño MVC (Model-View-Controller [9]), en particular, se realizó una aplicación Web, que se basó en el patrón anteriormente mencionado. Nos referimos en este trabajo a este prototipo como Sistema. En la Figura 4.1 se muestra un diagrama esquemático, mostrando como este patrón es aplicado en el prototipo. Por simplicidad, en la figura no se detallaron como parte del modelo[1] las clases del Modelo General de Juego y el Modelo del Usuario definidos en Capítulo 3.

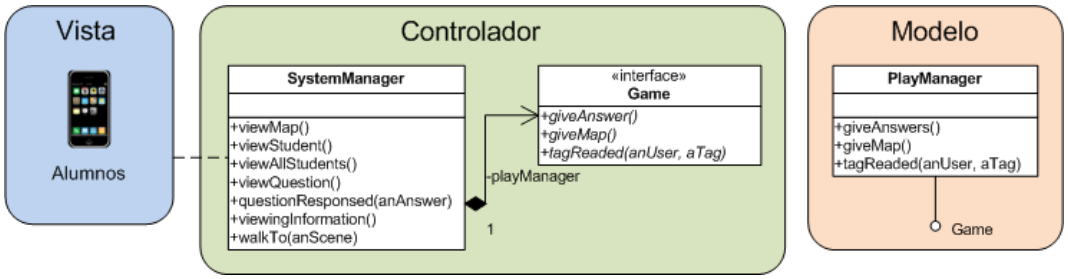

Figura 4.1: Prototipo implementado usando MVC.

Analicemos cada uno de los componentes involucrados en la Figura 4.1. Se puede apreciar que se destaca la clase *PlayManager* como parte del modelo (cabe recordar que dicha clase es la que se especificó en el Capitulo 2). En cuanto a la vista del Sistema, la misma consiste en páginas Web que le presentan al alumno la información o preguntas asociadas al juego.

El controlador se definió de una manera general para lograr interactuar con cualquier modelo de juego, para esta flexibilidad se definió la interfaz *Game*. Esta interfaz define el protocolo mínimo que debe tener cualquier modelo de juego para poderse usar. En particular, se puede apreciar que la clase *PlayManager* implementa esta interfaz.

[1] Tener en cuanta que se implementaron las clases presentadas en el Capitulo 2 junto con el comportamiento necesario para contar con la funcionalidad descripta en el Capitulo 2.

La clase *SystemManager* tiene el rol de controlador, logrando que el alumno pueda interactuar con el modelo definido recibiendo las páginas Web adecuadas. En las secciones que siguen se mencionaran los problemas detectados cuando se realizaron las pruebas prototípicas.

4.2. Monitoreo de los alumnos

Las pruebas prototípicas permitieron detectar la importancia de hacer un seguimiento (de ahora en adelante, monitoreo) de los alumnos y de ver el rol activo que cumple el docente. En esta sección se hará hincapié en lograr una solución para el monitoreo de los alumnos mediante el usó de un a herramienta.

Analicemos distintos monitoreos que podría realizar la herramienta que queremos brindarle a los docentes:

- *Reflejar solo la actividad actual de los alumnos.* Es decir, no se tiene conocimiento del resto de las actividades que conllevaron a que los alumnos estén en el estado actual. Es decir, los docentes no puede tener una idea global de cómo se fue desarrollando el juego, por ejemplo, determinar cuantas preguntas viene contestando bien cada alumno. Al final del juego, se pierde la idea de cómo trascurrió el juego, y cómo se desempeño cada alumno.
- *Reflejar toda la actividad una vez terminado el juego.* Es decir, al final del juego se tienen todas las actividades que realizaron todos los alumnos a lo largo del juego. Con lo cual primeramente será necesario almacenar esa información para luego poder acceder a ella y realizar el análisis correspondiente. Este tipo de monitoreo sirve para realizar análisis sobre el juego y las acciones de los alumnos pero no ayuda a al los docentes a determinar como transcurre el juego.
- *Reflejar la actividad actual y el historial de actividades.* Es decir, poder tener las dos opciones anteriormente reflejadas en forma combinada. Saber el estado actual de cada alumno, como así también las actividades que los mismos fueron realizando.

La última opción es la más rica para el docente, y se ajusta mejor al tipo de juegos que se plantean en este trabajo. Es decir, poder determinar las actividades actuales de los alumnos, como así también todo lo que vinieron realizando. Para que este monitoreo sea útil para los docentes, la herramienta debe contar con los

mecanismos necesarios para reflejar en tiempo real cada una de las actividades que realizan los alumnos. Poder realizar la observación en tiempo real de los alumnos de forma transparente (usando una herramienta) permite analizar el funcionamiento del juego, sin inhibir a los alumnos. Esto a su vez le da al docente la posibilidad de tomar acciones durante el juego para resolver problemas surgidos durante el mismo.

Analicemos diferentes situaciones dentro de un juego, destacando cómo una herramienta facilitaría la tarea de monitoreo de los alumnos, y permitiría al docente asistir a los mismos para que estos puedan completar el juego.

- Un alumno está respondiendo una pregunta, y que el tiempo estipulado máximo que tiene para hacerlo es de 5 minutos. Pasan 15 minutos y el alumno no responde. Una herramienta que permita monitoreo podría detectar esta situación y darle aviso al docente para que este se acerque al alumno y lo ayude para que pueda continuar el juego.

- Un alumno recibe un mapa que le indica donde está la siguiente pregunta. Pasados varios minutos el alumno no solo no llega al lugar correspondiente sino que visita otros que no lo corresponden. Sin un docente quien les pueda ayudar a comprender el mapa propuesto, este alumno puede que nunca llegue al lugar que le corresponde. Con una herramienta se le podría dar aviso al docente para que vaya a ver porque el alumno no llega al lugar indicado.

- Varios alumnos se encuentran jugando y algún evento hace que el lugar al que deben dirigirse se encuentra inaccesible (una habitación cuya puerta está cerrada con llave, por ejemplo). En este momento, la experiencia queda truncada. Sería deseable que ante este inconveniente, la herramienta detecte que el este alumno no avanza en el juego, y dirigirse al lugar donde se encuentra el mismo para ver porque no avanza en el juego.

A partir de estos ejemplos, se puede tener una idea más especifica de las facilidades que puede tener una herramienta que ayude al docente a monitorear este tipo de juegos. El docente tiene la responsabilidad de guiar al alumno durante el juego, es decir cuando los alumnos no entienden las preguntas realizadas o las informaciones dadas no le son claras. A su vez debe interpretar el comportamiento de los alumnos mientras ellos juegan para poder sacar sus propias conclusiones, y mejorar la experiencia de aprendizaje.

4.2.1. Herramienta de Monitoreo

Se han nombrado los beneficios de tener una herramienta para el monitoreo de los juegos, pensemos ahora cómo se puede diseñar esta herramienta. Si prestamos atención en los ejemplos dados a lo largo de las secciones anteriores, podremos observar que el monitoreo está centrado sobre los alumnos. Específicamente, lo que esta herramienta debe hacer es visualizar el estado completo del alumno, esto es: indicar dónde está, qué está haciendo, cuáles fueron las acciones anteriores que realizó, entre otros. De esta manera los docentes podrán ver dónde se encuentran los alumnos y qué están haciendo de manera fácil y simple. Toda esa información pertenece al modelo que se vio en el Capítulo 3, con lo cual la herramienta no estaría agregando información o comportamiento para realizar su tarea. Esto implica que mediante el modelo presentado previamente ya es suficiente y abarca las prestaciones requeridas de la herramienta. Para modelar la herramienta bastará con un conjunto de clases que observen al modelo y lo visualice de forma amigable para que el docente pueda comprender la información brindada y pueda tomar las decisiones que crea convenientes. Las clases que conformen la vista necesitarán actualizar en tiempo real la información mostrada ante los cambios que realicen los alumnos. Para ello las clases que realicen la tarea de monitoreo deberán observar al modelo y ante un cambio en él actualizarse.

La herramienta se construirá ampliando el controlador presentado en la Figura 4.1 con las clases y su comportamiento necesario para que los docentes puedan monitorear a los alumnos. La utilización de Seaside permitió realizar las extensiones necesarias para generar una nueva interfaz Web, con la cual el docente pueda visualizar la información de forma simple. Y esta nueva extensión puede funcionar en forma conjunta con el prototipo presentado en la Sección 4.1. La herramienta debe reflejar en tiempo real el estado de los alumnos, ya sea actual o historial de actividades, en particular, cada vez que cambia el estado de los alumnos, se debe ver en forma automática estos cambios en la pantalla del docente. Para poder proveer este mecanismo de actualización automática, se utilizó una implementación de Comet [8] para Smalltak y Seaside denominada Meteoroid [21]. Esta implementación permite de forma fácil y simple, configurar componentes Web que reacciona ante algún cambio en un modelo, y automáticamente los refleja en la interfaz. Esto se mostrará con más nivel de detalle en el Capítulo 5.

La Figura 4.2 muestra como se amplió el controlador. Se puede observar que la clase *SystemManager* agregó el comportamiento especifico para poder proveer monitoreo a los docentes. Como hemos mencionado anteriormente es necesario almacenar la actividad de los alumnos, esto recae sobre la clase *DBManager*, la cual permite almacenar cada una de las actividades de los alumnos.

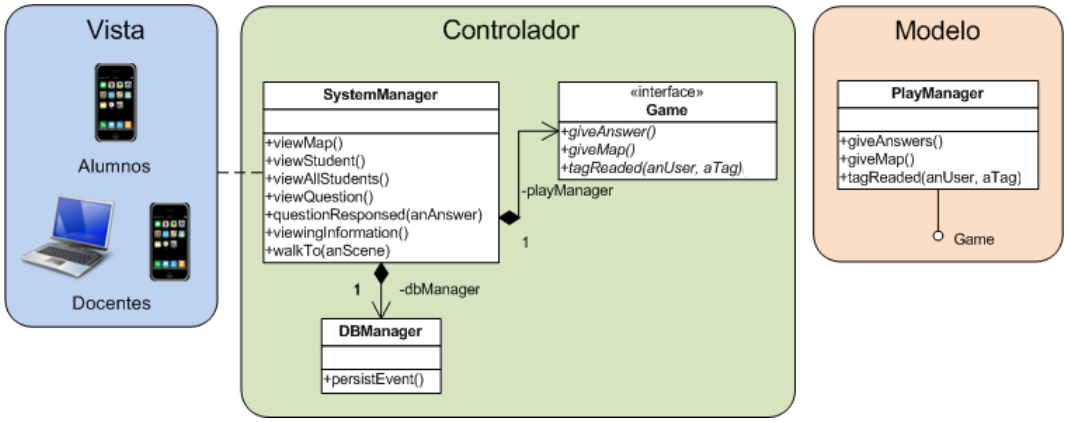

Figura 4.2: Monitoreo de los alumnos.

En la Figura 4.2, a su vez podemos observar que se agregó la vista de los docentes. En el caso de los alumnos, se seguirá mostrando la información correspondiente al lugar donde haya llegado, como una pregunta o información y los mapas que se les irán entregando para dirigirse de una escena a otra. En cambio, la vista del docente debe reflejar todo el comportamiento de todos los alumnos, es decir, va a permitir el monitoreo de los mismos. Se tendrán diferentes aspectos a monitorear de los alumnos: su posición, su actividad, su estado y además poder tener una vista general de todos los ellos. En el Capítulo 5, se mostrará la implementación concreta del monitoreo.

4.3. Sistema de Recuperación

Como se mencionó al comienzo de este capítulo, durante las pruebas realizadas ocurrieron varios problemas los cuales dificultaron notoriamente la experiencia del juego. Con estos problemas ocurridos, nos dimos cuenta que era necesaria una herramienta que ante estas eventualidades, pudiera realizar los cambios o ajustes necesarios para que el juego pueda terminar de manera satisfactoria. Para ello es necesario primero comentar cuáles son los problemas que hacen necesario un sistema de recuperación. Un sistema de recuperación ideal debe dejar al sistema en un estado consistente, acorde al estado anterior que éste tenía antes de que ocurriera la falla. Para ello hay que observar qué elementos tendremos que tener en cuenta para que la recuperación se pueda llevar a cabo. Es decir qué elementos del modelo se verán involucrados y si el modelo propuesto en el Capítulo 3 es suficiente o no para poder hacer la recuperación.

Hemos visto que existen varias problemáticas que afectan el monitoreo de

los juegos educativos, ahora veremos que esas mismas problemáticas afectan al buen desarrollo de los mismo estropeando la experiencia. Existen distintos tipos de problemas, algunos recurrentes en la Computación Móvil, otros específicos del dominio de los juegos y otros correspondientes a las acciones que los alumnos realizan durante el juego.

4.3.1. Interacción de los alumnos con el juego

Uno de los problemas observados durante las pruebas realizadas es que los alumnos no siempre completaban las acciones solicitadas en la experiencia, esto se daba por la falta de comprensión de la dinámica del juego o de la tecnológica o porque simplemente decidieron seguir su propia dinámica. Es decir, que por más que antes de jugar se hiciera una explicación del sistema a los alumnos que formaban parte de la prueba, ellos realizaban otras acciones o realizaban las órdenes en diferente orden. La cuestión principal era que, como el Sistema no se encontraba preparado para soportar el uso de acciones inesperadas o la incompletitud de las acciones y, por ende, el Sistema fallaba. Pero esto fue una falencia en el prototipo (Sección 4.1).

Es momento de analizar las acciones y ver de qué manera es posible evitar que el Sistema falle por realizarlas de manera diferente a la esperada. Es de notar que muchas de esas acciones son las comunes que cualquier alumno puede realizar en el transcurso del juego. Por ejemplo:

- El alumno cerró el navegador Web.
- El alumno lee repetidas veces el mismo código de barras 2D.
- El alumno lee un código de barras 2D que no es el que se le había indicado.
- El alumno lee un código de barras 2D, el sistema le entrega una pregunta para contestar y en lugar de hacerlo se mueve y lee otro código de barras.
- El alumno tiene una pregunta para contestar y pulsa varías veces el botón de Enviar Respuesta.
- El alumno tiene una pregunta para responder, contesta y el sistema le otorga un mapa. El alumno utiliza el botón de Volver del navegador, cambia la respuesta y envía nuevamente la respuesta al sistema.

Estas son situaciones normales y que se suceden en los juegos, por lo tanto el modelo de los mismos debe controlar este tipo de anomalías. Es decir, son eventualidades y mal uso del Sistema y se pueden solucionar sin la necesidad de tener una herramienta aparte que resuelva los inconvenientes producidos. En realidad el mismo Sistema debe tener alguna manera de identificar estas acciones y continuar con el juego sin interrumpirlo. El tipo de interacciones vistas en las pruebas fueron comunes en diferentes tipos de grupos de alumnos, con lo cual el modelo usado para estas aplicaciones debe tener en cuenta las acciones descriptas dado que son fácilmente solucionables. Como ya mencionamos anteriormente nosotros hicimos las pruebas con el modelo descripto en el Capitulo 3 pero se podría tener otro modelo siempre y cuando el mismo respete la interfaz *Game*.

En conclusión, el modelo de juego debe ser el encargado de contar con la lógica necesaria para aquellas anomalías surgidas de una interacción que hacen los alumnos aun cuando esa interacción no es la esperada. Y no es necesario tener una herramienta para este tipo de fallas.

4.3.2. Problemas técnicos

Existen también problemas técnicos ajenos a la interacción usuario/Sistema, por ejemplo en el dispositivo. En muchos de los casos que nos surgieron en las pruebas, la solución fue cambiar de dispositivo dado que el que se estaba usando quedaba inoperable. Con el cambio de dispositivo se pueden resolver una gran gama de problemas, pero pensemos como debería hacer el Sistema para reconocer este inconveniente y detectar que un alumno ha cambiado de dispositivo. Es decir, el Sistema debe tener conocimiento de este cambio de dispositivo dado que debe entregar de manera correcta las informaciones a los alumnos que las hayan solicitado. Utilicemos un nuevo ejemplo para ilustrar esta situación y distinguir mejor el problema. Usemos como escenario el visto en el Capítulo 2 donde existen 3 alumnos. Supongamos que el Jugador A se encuentra en la escena *E1.1* y luego de responder la pregunta tiene un problema con su dispositivo, el cual deja de funcionar. El Jugador A cambia el dispositivo por otro. Al realizar el cambio, el Jugador A no tiene el mapa que le debe proporcionar el sistema para continuar con el juego. El problema radica en que el sistema no puede distinguir que el Jugador A cambió de dispositivo sin que este se lo informe. El sistema no puede enviar ningún dato al dispositivo del Jugador A dado que no tiene forma de identificar al alumno. Esto es un gran problema porque invalida la posibilidad de cambiar de dispositivo cuando este falla. Para solventar este inconveniente el sistema debería tener alguna forma de poder identificar a los alumnos. Esto genera un nuevo problema: de identificación de los alumnos y de sus dispositivos. En esta situación hay que proveer los mecanismos necesarios para que el alumno

pueda continuar el juego desde otro dispositivo. Para proveer identificación, los navegadores Web utilizan sesiones entre el cliente y el servidor. Con esta estrategia es posible identificar a los alumnos.

Analicemos otra situación, ¿qué pasa si el dispositivo se apaga o se cierra el navegador?, ¿habremos perdido la forma de identificar al alumno? Dependiendo del navegador Web y de la circunstancia ocurrida se puede perder la identificación del alumno. Con lo cual es necesaria una herramienta que nos ayude a resolver este problema.

Tengamos como ejemplo la situación mostrada en la Sección 2.6 donde teníamos a un alumno jugando, ¿qué ocurrirá si en medio del mismo el dispositivo se apaga? ¿Qué ocurrirá con los datos del alumno? ¿El alumno podrá seguir jugando? ¿De qué manera continuará el juego? ¿A partir de qué parte del juego continuará? Todas estas interrogantes deben ser respondidas, el alumno no puede quedar sin completar el juego por una razón de tipo técnica o tecnológica como en este caso. Para algunas problemáticas, la solución alcanza con el cambio de dispositivo, mientras que para otras será necesario el uso de una herramienta que dispongan los docentes, la cual realice los cambios necesarios en el modelo para que se pueda continuar y finalizar el juego. Tengamos en cuenta otro posible escenario. Durante el juego se corta la luz. Para los alumnos este inconveniente puede pasar inadvertido dado que al utilizar dispositivos móviles estos poseen sus propias fuentes de energía, pero en el caso del servidor estos datos se perderían dejando al Sistema en un estado inconsistente. Una vez que el suministro eléctrico retorne los alumnos no podrán continuar porque todo su avance habrá desaparecido, ya que las secciones en el servidor se borraron. Esto requiere de recuperar (por ejemplo, de una base de datos) todas las actividades realizadas por los alumnos, y volver a dejar todos los estados consistentes. Los alumnos deberán volver a ingresar al sistema, para que el mismo le informe como continuar el juego. Esta solución también serviría si se quiere continuar el juego en otro momento sin la necesidad que los alumnos tengan que realizar todas las acciones de nuevo. Usando los datos almacenados de los alumnos, se podría continuar el juego en cualquier momento, donde cada alumno continúa el mismo acorde a la última que había realizado.

A continuación se hace un listado a modo de resumen de los problemas detectados en las pruebas, relacionados con los problemas técnicos:

- Pérdida de señal de WiFi.
- Reinicio del dispositivo de forma automática, por ejemplo por falla del SO.
- El dispositivo queda inoperante por algún motivo. Esto conlleva a reiniciar

el mismo.

- El dispositivo solo posee pantalla táctil y ésta deja funcionar.
- Apagado del dispositivo por batería baja.
- Se vence la sesión del alumno en el navegador Web que está utilizando para el juego.
- El servidor presenta algún problema y se debe reiniciar el mismo.

4.3.3. Herramienta de Recuperación

Hemos visto, en la Sección 4.2.1, que para tener una herramienta de monitoreo no es necesario agregar comportamiento al modelo propuesto en el Capítulo 3, pero en cambio, para la recuperación del sistema, será necesario que el modelo provea cierto comportamiento que previamente no tenía. A continuación se listan los aspectos que el modelo debe agregar para poder generar una herramienta que satisfaga la recuperación del sistema ante fallos:

- Manejo de sesiones de los navegadores Web.
- Identificación de los alumnos.

Con la funcionalidad mínima requerida será posible diseñar una herramienta para la recuperación del sistema ante los problemas planteados. Esta funcionalidad debe recaer sobre el modelo que implementa los juegos dado que sin ella no sería posible siquiera jugar.

Supongamos que durante un juego acontece uno de los problemas mencionados en la Sección 4.3.2 y se debe restablecer el estado del alumno. Para mantener la misma idea de reflejar todas las acciones involucradas con el alumno, será necesario agregar una acción que sea la de restablecimiento. Con esta acción estaremos mostrando dentro del modelo que durante el juego, por alguna eventualidad, fue necesario utilizar la recuperación del sistema y ello quedó reflejado en el historial de acciones del alumno en cuestión.

Además, será necesario agregar un estado más al planteado en el modelo del Capítulo 3. Este nuevo estado debe representar que la información de un alumno está siendo restablecida y que sus acciones se están ejecutando nuevamente para

dejarlo listo para continuar. Este estado juega dos papeles importantes, el primero es que refleja la actividad del alumno en todo momento, incluso cuando se encuentra recuperando las acciones y en segundo lugar (pero no menos importante) evitar que, por la recuperación, se genere un lazo infinito. Esto se debe a que el hecho que un alumno cambie de estado, éste debe ser almacenado y eventualmente recuperado. Pero si durante la recuperación se vuelve a ejecutar una acción que implica recuperar las acciones, se estará entrando en una ejecución sin fin haciendo que no se pueda volver a jugar.

Por otra parte hemos dicho que la implementación del modelo está basada en una aplicación Web, ahora será necesario que de alguna manera se lleven las verificaciones necesarias para el manejo de sesiones de los navegadores Web. Con esto solucionamos ciertos comportamientos de los alumnos con el Sistema, por ejemplo, cerrar el navegador Web ya sea por equivocación o a propósito. Será necesario mantener algún mecanismo que permita mantener la conexión entre el servidor y el dispositivo móvil para el envío de información. Esto también afecta a la identificación de los alumnos, dado que como se ha mencionado anteriormente, el cambio de dispositivo puede ser la única solución posible para sortear algún inconveniente. El sistema debe ser capaz de relacionar al alumno con el nuevo dispositivo de manera que pueda seguir jugando y que la información que le envíe el servidor sea la correcta.

La Figura 4.3 ilustra los agregados que son necesarios para que el modelo pueda recuperarse ante fallas. En la figura se ve el modelo planteado anteriormente. Podemos observar que dentro del modelo se agregaron las clases necesarias para reflejar el cambio de estado del alumno al momento en que se encuentra en recuperación (*Restoring*) y el evento que refleja el acto de restablecer al alumno ante alguna eventualidad (*Restored*). Por el lado del controlador, se puede ver que se ha agregado la clase *Restorer* la cual tiene la responsabilidad de restaurar el sistema en su totalidad o a algún aspecto en particular (un alumno, un juego o una puesta en escena). También se ve que dentro de la clase *SystemManager* se han agregado los mensajes necesarios para realizar la restauración del sistema.

Si contásemos con la herramienta de monitoreo de la cual se habló en la Sección 4.2.1, un docente ante algún error del sistema (por ejemplo, que brindó un mapa que no correspondía, o por simple mal funcionamiento) podría restaurar el estado del alumno a uno en el cual fuera consistente. De esta manera el alumno no se enteraría del cambio ocurrido y podría continuar tranquilamente el juego. También se podría detectar un problema en el juego y solucionarse sin que los alumno se enterasen. Con esto queremos decir que con ambas herramientas la cantidad de situaciones que se pueden solucionar es muy bastante amplia.

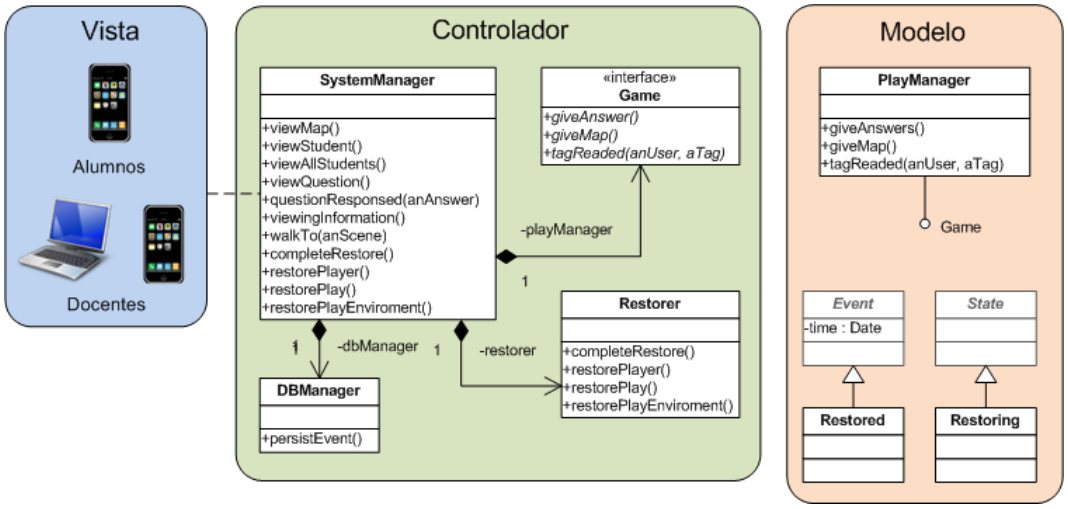

Figura 4.3: Sistema de Recuperación y Monitoreo de los juegos.

A lo largo de este capítulo hemos visto la importancia de tener dos herramientas que nos ayuden tanto a ver y controlar el comportamiento de los alumnos, como tener un sistema que ante cierto tipo de fallas se pueda recuperar y que el juego se pueda terminar de manera correcta. Ambas herramientas influyen en el desarrollo del juego, pero su implementación no debe afectar al modelo propuesto en el Capítulo 3 de forma sustancial. Solamente contar con los estados y eventos relacionados con la recuperación.

Capítulo 5

Herramienta

En este capítulo se mostrará y describirá la herramienta que ayudará a los docentes en el monitoreo y en la recuperación en los juegos educativos móviles basados en posicionamiento. Este capítulo estará divido en dos secciones. En la primera se describirá cómo se implementó la capa del controlador que se mencionó en el Capítulo 4. Se contará las decisiones de diseño y como son los componentes para realizar las tareas de monitoreo y de recuperación. Además se describirán las características que posee la herramienta junto con las problemáticas que resuelve. En la segunda sección, se describirán las diferentes interfaces (vistas) que posee la herramienta explicando cómo funciona y la relación existente con los componentes del controlador y cómo se conectan con el modelo propuesto.

5.1. Capa del Controlador

En esta sección y en sus subsecciones veremos cómo está implementado el controlador y cómo hace para poder realizar las tareas tanto de monitoreo y de recuperación. Utilizaremos esta herramienta para mostrar los lineamientos a seguir para generar herramientas del mismo tipo, es decir que sirvan para el monitoreo y recuperación ante fallas de las aplicaciones educativas basadas en posicionamiento ya que el modelado de los conceptos relacionados a los juegos puede variar.

5.1.1. Monitoreo

En esta subsección mostraremos los mecanismos necesarios para poder brindar el monitoreo del modelo planteado. En particular, se mostrará el monitoreo desde tres puntos de vista diferentes, los cuales pertenecen a: los jugadores, los ambientes de juego y por último a los juegos.

Monitoreo del Jugador

Hemos hablado que para monitorear al Jugador (o al alumno como es en nuestro caso en particular) debemos contar con una implementación del patrón de diseño *Observer* [9]. En la implementación debemos tener como sujeto a observar al jugador, y como observador al componente del controlador que le indicará a la vista cómo y cuándo se debe actualizar. Para ser más precisos, debemos tener un controlador por cada jugador que debe observar el cambio de estados del mismo. Es necesario que el modelo cuente con los mecanismos necesarios para avisar de todos los cambios que ocurran, de esta forma, el controlador podrá capturar el cambio y reflejarlo en la vista correspondiente. Para ello, en la implementación generada se utilizó el sistema de dependencias que posee el ambiente elegido para el desarrollo. Para clarificar este concepto nombremos al controlador del jugador como *PlayerController*, el cual debe agregarse a la capa del controlador. Cada *PlayerController* debe conocer a un *Player* al cual observar y la correspondiente vista para actualizarla. Este controlador será el responsable de actualizar la vista del jugador en cada uno de los aspectos necesarios para tener un monitoreo eficiente y en tiempo real. La Figura 5.1 muestra la relación entre la clase *Player* y el *PlayerController*. Como hemos mencionado al comienzo de este capítulo, para que los controladores funcionen, será necesario que el modelo implemente algún sistema de notificación. Para este caso particular, la clase *Player* deberá implementar el sistema de notificaciones en el cambio de estados y de atributos del jugador.

La Figura 5.2 muestra, con un diagrama de secuencia, como se modifica el estado en un Jugador luego de leer un código 2D, y cómo es capturado por el controlador avisándole a la vista del docente y del alumno que se actualicen.

Hemos dicho en capítulos anteriores que conocer el estado actual de un jugador es importante para llevar a cabo el monitoreo, pero no es suficiente. Será importante conocer el historial del mismo, para que el docente o administrador de los juegos pueda realizar el seguimiento de mejor manera y con mayor cantidad de datos, para comprender el comportamiento del jugador. Para ello será necesario

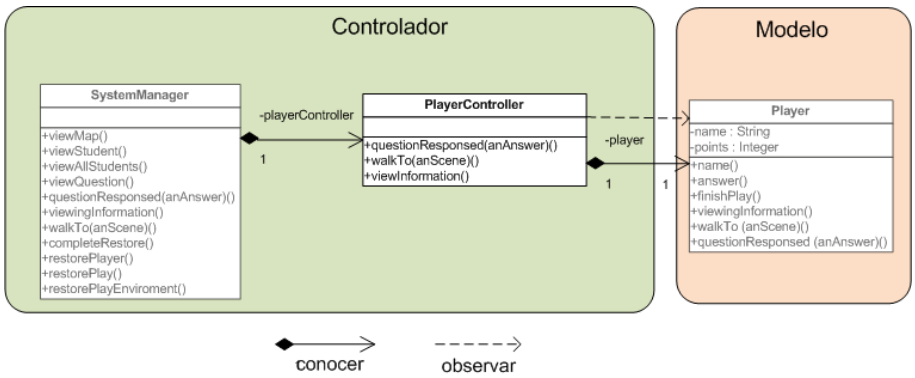

Figura 5.1: Clase PlayerController.

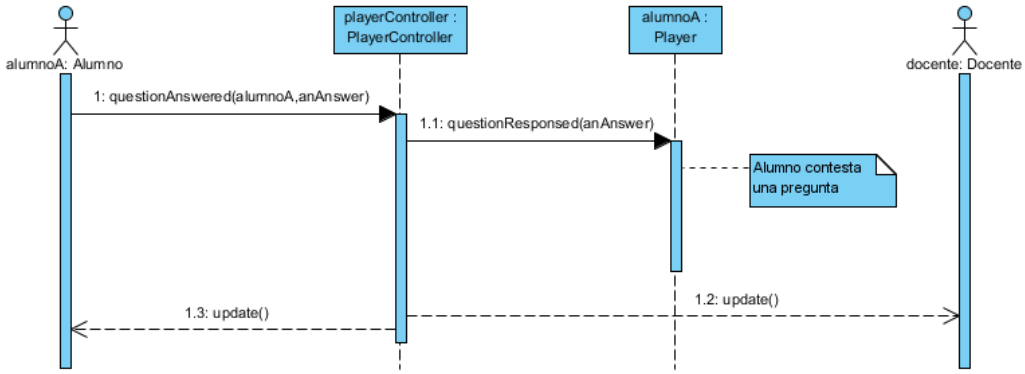

Figura 5.2: Cambio de estado capturado por el controlador.

mostrar el historial de eventos del jugador, al docente. En el modelo planteado en el Capítulo 3, se tiene el historial de cada jugador almacenando los eventos que fue realizando el jugador. Nuestro *PlayerController* deberá observar además del estado, el historial del mismo. De la misma forma que antes, será necesario agregar la funcionalidad necesaria para que, ante el agregado de un nuevo evento al historial, el controlador pueda enterarse del cambio y avisarle de forma precisa a la vista de cómo actualizarse. Una vez más, utilizaremos el patrón *Observer* para resolver esta problemática.

Conocer tanto el estado actual como el historial del jugador, alcanzará para que el docente pueda visualizar todos los aspectos necesarios para realizar el monitoreo en tiempo real deseado.

Monitoreo del Ambiente de Juego

De la misma forma que resulta importante realizar el seguimiento de un jugador en particular, será interesante y útil poder tener un seguimiento más global de los jugadores que se encuentran jugando un mismo juego en un mismo lugar. Así como hemos nombrado en el Capítulo 3 que es necesario modelar el ambiente de un juego en particular, ocurrirá lo mismo a la hora de realizar el monitoreo. Será necesario tener un controlador específico por cada ambiente de juego (*PlayEnvironment*) que se tenga. Este controlador, llamado *PlayEnvironmentController* deberá observar los posibles cambios que pueda tener un *PlayEnvironment*. De la misma forma que hemos hecho con el jugador, aplicaremos la idea sobre los *PlayEnvironment*. El controlador, en este caso, observará cuando un jugador se suma y abandona el juego. El resto de las características, al no poder ser alteradas (dónde se realiza el juego y cuál es el orden de las escenas) no será necesario observarlas continuamente. Con lo cual este controlador es de fácil implementación. La Figura 5.3 muestra el agregado de la clase *PlayEnvironmentController* a la capa del controlador. De esta forma, cuando un jugador se una (o finalice un juego), el docente podrá verlo en tiempo real. La Figura 5.4 muestra un diagrama de secuencia de la situación descripta anteriormente, es decir cuando un jugador se une a un juego y cómo el controlador captura esa acción y la refleja en la vista del docente.

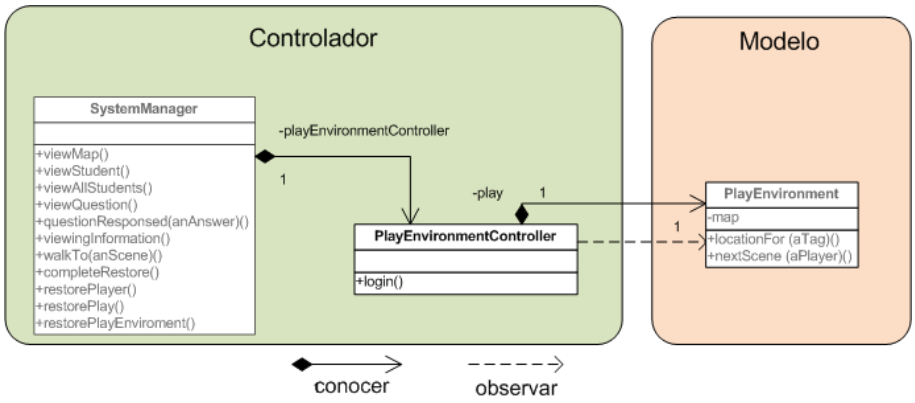

Figura 5.3: Clase PlayEvironmentController.

Por otro lado será importante poder saber qué escena se lleva a cabo en cada lugar. Mediante este controlador, será posible obtener esta información y visualizarla de manera conveniente para que aquella persona (por ejemplo un docente) que utilice la herramienta le sea útil y fácil de utilizar. Con esta información, y con las ubicaciones de todos los jugadores que se encuentran jugando se podrá representar gráficamente la disposición de los mismos. De esta manera se podrá tener una idea global con respecto a todos los participantes en todo momento.

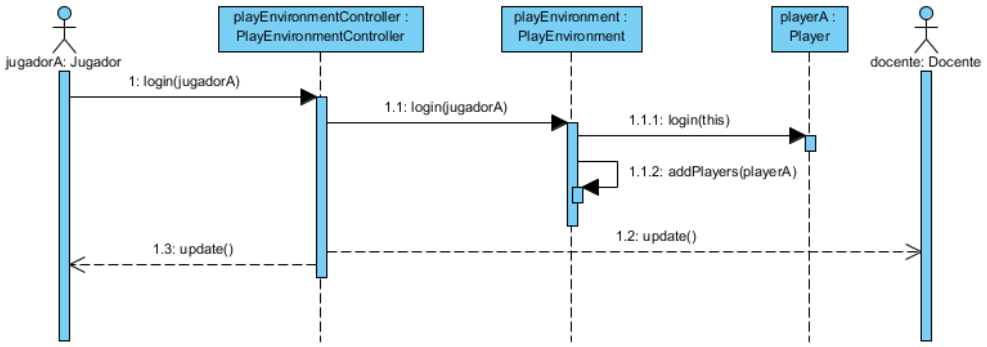

Figura 5.4: Un docente monitorea un juego y se une un jugador.

Monitoreo del Juego

Al igual que los jugadores y los ambientes de juego, es importante tener los datos de los juegos en general. Para el alcance de este trabajo, tanto los ambientes de juego como los juegos poseen una variabilidad casi nula. Esto se debe a que una vez instanciado un juego no es posible modificarlo. Pero es conveniente poder acceder a los datos del mismo para que el docente o administrador del sistema los pueda ver. El controlador para los juegos es el *PlayController* y se modeló para que en un futuro, si es necesario agregarle comportamiento, pueda realizarse de forma fácil y simple. La Figura 5.5 muestra esta nueva clase en la capa del controlador.

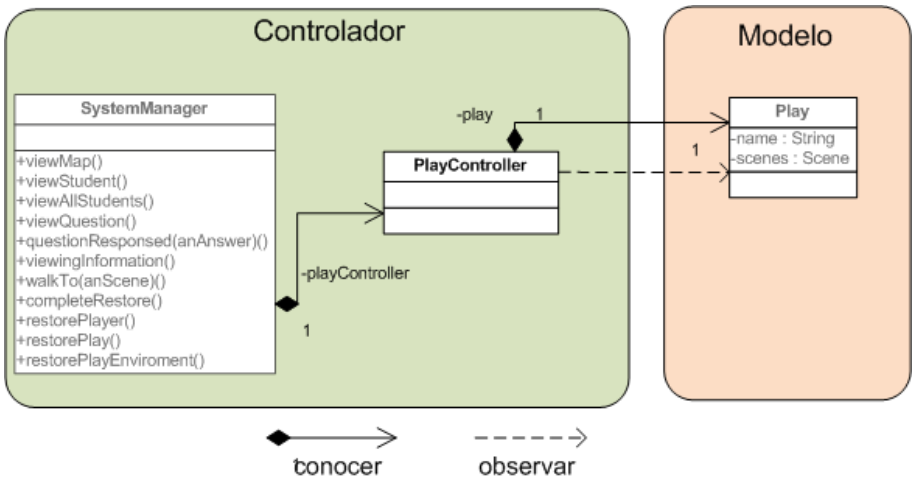

Figura 5.5: Clase PlayController.

De esta manera se respeta en el *MVC* en los diferentes aspectos de los objetos involucrados (los jugadores, los ambientes de juego y los juegos) y se deja la

posibilidad de extender su comportamiento de forma simple. Por último la Figura 5.6 muestra la capa completa del controlador que se ha descrito en esta sección.

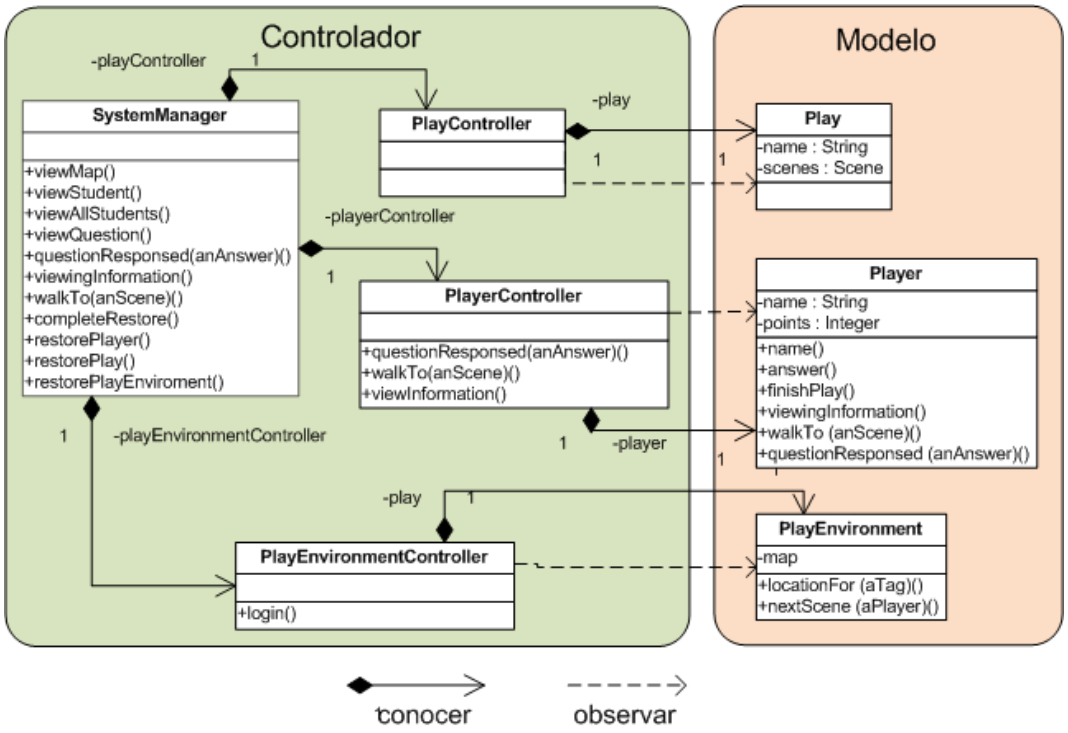

Figura 5.6: Capa completa del controlador.

5.1.2. Recuperación

Hemos mencionado en la Sección 4.4.5 distintos casos y ejemplos donde sería necesario tener un sistema de recuperación. A diferencia del monitoreo, la recuperación del sistema ante fallas es más complejo. En principio se debe contar con alguna clase que realice la persistencia de la información y otra (eventualmente la misma) que recupera la información persistida. En el modelo de solución propuesto en el Capítulo 4, hemos nombrado a las clases *DBManager* y *Restorer* para cumplir con los roles antes mencionados. La implementación del *DBManger* podrá variar dependiendo del tipo de persistencia utilizada, es decir si es utilizando archivos de texto plano, utilizando una base datos relacional o no, utilizando archivos *XML*, entre otros. Lo importante es que esta clase pueda tomar los datos de los distintos objetos y pueda persistirlos. De forma análoga será conveniente que esta clase (u otra) pueda realizar la tarea inversa, es decir recuperar los datos previamente persistidos y convertirlos en objetos del dominio.

La clase *Restorer*, en particular, tiene la responsabilidad de tomar los datos

obtenidos por el *DBManager*, y realizar las acciones necesarias para restaurar el sistema y dejarlo en plenas condiciones para continuar con el juego para que pueda ser finalizado de forma correcta. Queda por ver cómo podremos hacer para restaurar el sistema. La primer idea que surgió, fue la de utilizar los eventos persistidos, convertirlos a objetos del dominio y reemplazarlos sobrescribiendo los que posee el jugador al cual se quiere restaurar. Esta idea de fácil implementación, no funcionó, dado que si por alguna razón algún evento persistido tenía algún error este se vería reflejado nuevamente en el jugador luego de haber sido restaurado. Es decir, que no se cumpliría el objetivo mismo de la restauración dado que el jugador quedaría con datos erróneos al igual que antes de la restauración.

La solución puesta en práctica, radica en que en vez de sobrescribir los eventos persistidos por los actuales es mejor ejecutarlos nuevamente. Ya hemos dicho en el Capítulo 3, que cada evento posee la información necesaria para describirse. Ahora será necesario utilizar esta información para simular que el jugador se encuentra jugando una vez más el juego, pero realizando las acciones en el orden en el cual fueron persistidas. Para poder realizar esta estrategia, y para no agregar comportamiento al modelo propuesto, se debe averiguar por cada evento de qué tipo es. Para esto se puede utilizar el patrón de diseño *Visitor* [9]. El *Restorer* en el mensaje accept deberá enviarse como parámetro y cada evento le responderá de qué clase es mediante un mensaje, el cual siguiendo los lineamientos del patrón, deberá ser visit seguido el nombre del evento. La responsabilidad del *Restorer*, será la de utilizar los datos del evento y realizar las acciones convenientes para reflejar ese comportamiento en el jugador. Vemos su modo de uso en la Figura 5.7, donde el *Restorer* primero borra el historial del jugador (llevándolo a un estado consistente), luego le pide al *DBManager* todos los eventos persistidos y los convierte en objetos del dominio. Seguido a esto, visita uno a uno pasándose a sí mismo como parámetro para que cada evento le responda cómo debe visitarse.

De esta manera es posible reproducir todos los eventos por los cuales pasó un jugador, y volver a tener una instancia consistente del mismo, permitiendo de esta forma poder completar el juego.

Con esta idea, la recuperación del sistema se hace de forma fácil, dado que como lo muestra la Figura 5.7 lo que se debe hacer es tomar todos los eventos persistidos y hacer que cada uno se ejecute nuevamente. Cada evento en particular le dirá al *Restorer* qué acciones y mensajes enviarle al jugador para que quede reflejado que se ha realizado ese evento. A continuación, se muestra, en la Figura 5.8 un caso particular de evento recuperado (*StartPlay*), y como se puede ampliar lo que hace el *Restorer* como respuesta al visit.

Ahora veamos en la Figura 5.9 cómo es el comportamiento a la hora de recuperar un evento de llegada a una escena (*Arrive*).

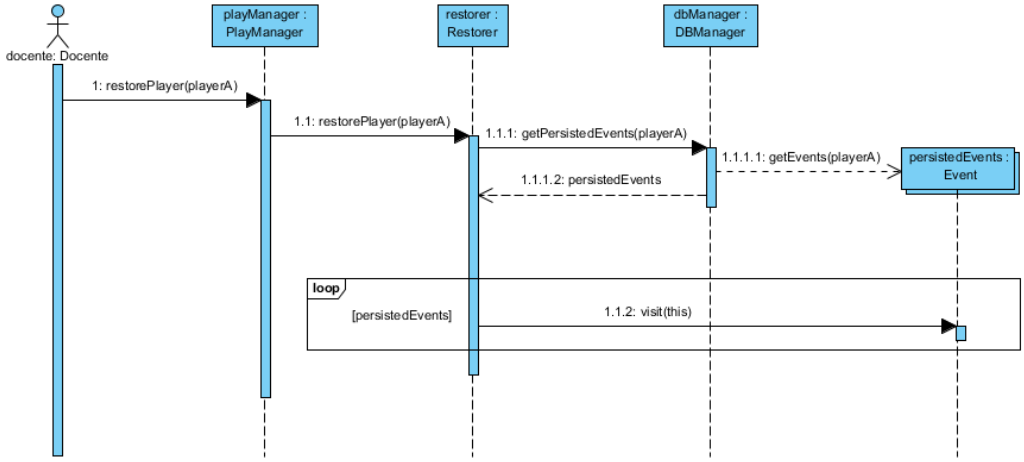

Figura 5.7: Recuperación de los eventos persistidos.

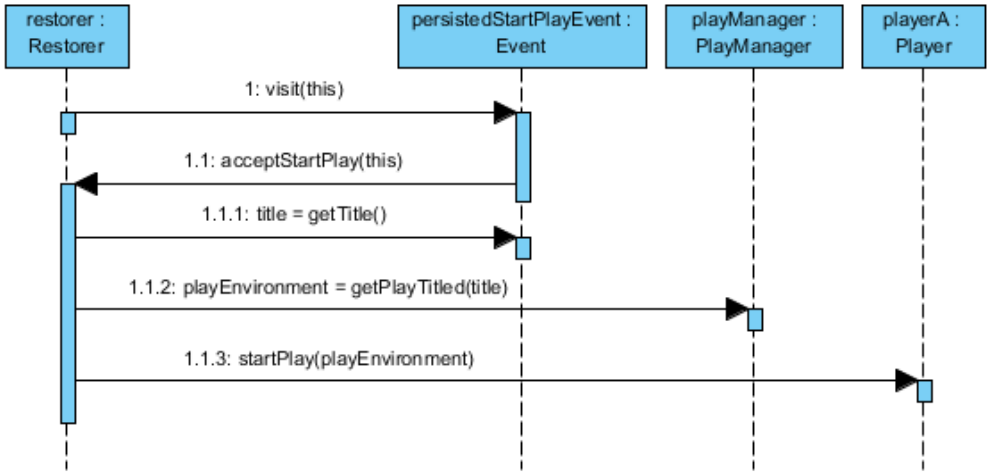

Figura 5.8: Recuperación del evento *StartPlay*.

Como se ha mencionado en el Capítulo 4, es importante tener en cuenta un caso en particular cuando se está haciendo la recuperación. También al recuperar a un jugador se creará un evento que refleje esta acción. Esto implica que al finalizar la recuperación, será necesario crear un nuevo evento *Restored* al jugador. De esta manera, tendremos reflejado todos los eventos que tuvo el jugador. Hasta el momento, los eventos del jugador eran generados por él mismo, pero ahora se agrega uno externo que es de la recuperación. Será de igual importancia saber en qué momento fue recuperado el jugador, con lo cual reflejar esa acción como un evento mantiene la coherencia en el modelo planteado. Ahora, si tuviésemos que recuperar a un jugador cabe preguntarse qué efecto deberá tener un evento como el *Restored*. No tendrá sentido, mientras se está recuperando a un jugador,

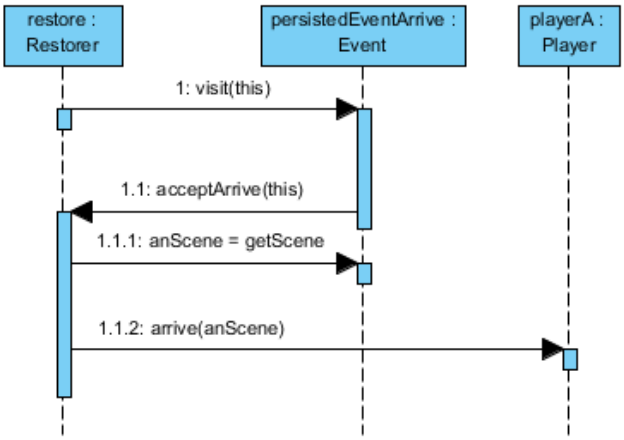

Figura 5.9: Recuperación de un evento *Arrive*.

volver a recuperarlo, hacerlo implicaría quedar en un lazo infinito. Por lo tanto las acciones que debe realizar el evento *Restored* son las de reflejar que ha sido recuperado, es decir que simplemente se debe agregar el evento en el historial del jugador. La Figura 5.10 muestra el diagrama de secuencia con el cual un evento *Restored* es ejecutado. Podemos ver que simplemente le indica al jugador que debe crear un nuevo estado de recuperación y almacenarlo.

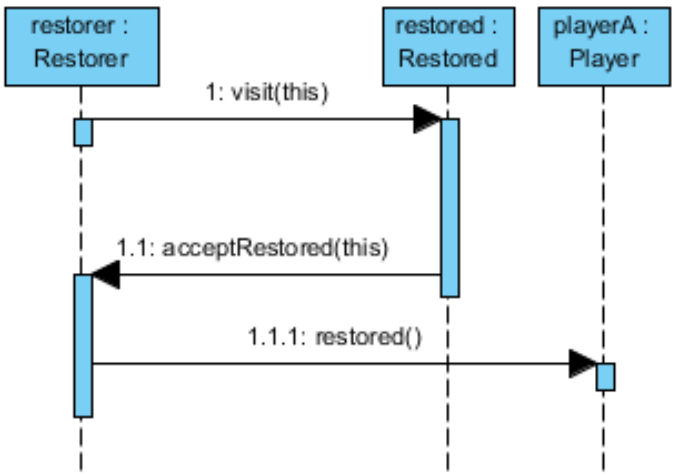

Figura 5.10: Recuperación del evento *Restored*.

Una vez resuelta la recuperación de un jugador en particular, es posible escalarlo para un número cualquiera de jugadores. Es decir que podremos recuperar a todos los jugadores que estén jugando en un momento dado. Esto nos permite (entre otras tantas opciones) pausar el juego y continuarlo en otro momento. De

esta manera recuperar el juego por completo representa recuperar a todos los jugadores.

5.2. Capa de la Vista

En esta segunda sección, abordaremos las diferentes vistas que posee la herramienta implementada y cómo es la relación entre cada uno de los elementos visualizados, junto con sus correspondencias al modelo o al controlador. Mostraremos dónde se efectúan cada una de las funcionalidades previstas.

Veremos a continuación las opciones que se diseñaron para la herramienta. En las siguientes figuras iremos viendo las distintas pantallas con su propósito y el repertorio de acciones y opciones que cuenta cada una. Luego se mostrará la herramienta en uso con un alumno jugando y luego varios alumnos jugando el mismo juego.

Luego de una identificación y autenticación en el sistema, la herramienta muestra la pantalla que se puede observar en la Figura 5.11. En ella se puede apreciar que existen diferentes secciones discriminadas. Entraremos en detalle para cada una de ellas. Estas son cinco secciones:

- General(General): Aquí se encuentra la funcionalidad para salir de la herramienta y a su vez, muestra la iconografía utilizada en ella a modo de ayuda.
- Juegos(Plays): En esta sección se puede ver la información relevante a los juegos, es decir a su estructura interna, la cual como ya se ha dicho anteriormente no varía en sus diferentes realizaciones.
- Ambientes de Juegos(Plays in Action): Se muestra la información relevante a las puestas en escena, lugar donde se lleva a cabo, qué alumnos se encuentran participando, distribución de la información del juego, entre otros aspectos.
- Jugadores(Players): Se muestra la información detallada de todos los jugadores en general y en particular al juego que están jugando.
- Administración(Administration): Información referente al administrador (o docente) sobre los mecanismos para guardar los datos dentro de la herramienta.

Figura 5.11: Pantalla con el menú de opciones principales del docente.

5.2.1. Iconos de la herramienta

Veamos en detalle la iconografía utilizada para poder comprender mejor las siguientes secciones de la herramienta. La Figura 5.12 muestra todos los íconos creados junto con sus significados. Se puede ver también que se han categorizado y dividido en 5 grupos. Esta división hace que sea más fácil comprender el significado del icono y asociarlo a un conjunto semejante de acciones o estados que le son afines a cada icono. Estos grupos son:

- Estados del Jugador (Player states).
- Eventos del Jugador (Player events).
- Atributos del Jugador (Player Attributes).
- Acciones del sistema del Administrador (Admin system Actions).
- Grupos de jugadores (Players Groups).

En la primer columna, se tienen todos los posibles estados en los cuales un jugador puede estar. Recordemos que un jugador es un alumno o un grupo de

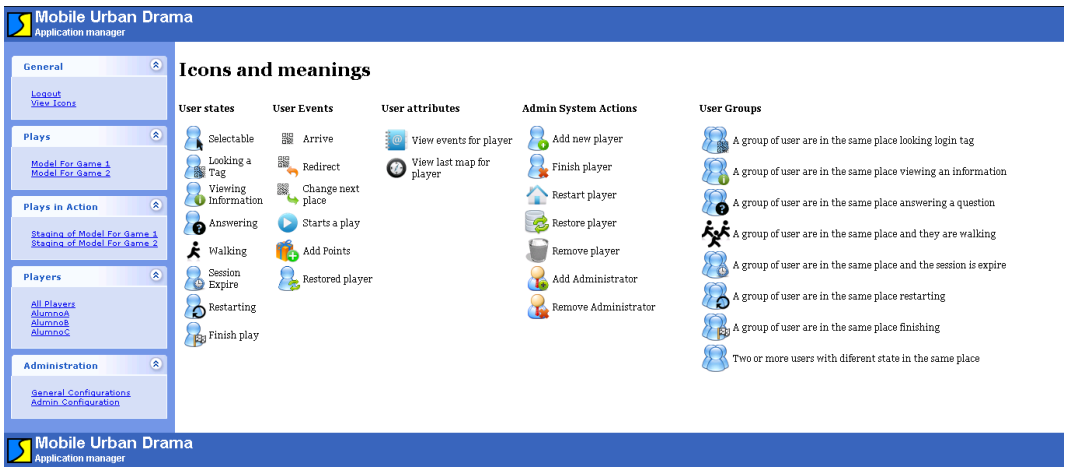

Figura 5.12: Pantalla con los íconos y sus descripciones dentro de la herramienta.

alumnos. Es de notar que la cantidad de estados que hay, es mayor a la propuesta en el Capítulo 2. Esto ha sido así, dado que resultó útil tener más estados (*Restarting* y *SessionExpire*) que en el modelo planteado, para brindar más precisión en la información brindada. En este caso, para esta herramienta, creímos conveniente incorporar más estados que reflejen mejor las actividades de los alumnos mientras juegan. Los estados que se tienen en cuenta son los que se muestran en la Tabla 5.1.

Como se vio en el Capítulo 4, los eventos son los generados cuando los jugadores realizan ciertas acciones que nos son importantes. Estos eventos son los mostrados en la Tabla 5.2.

Los atributos de los jugadores son accesos a información específica del jugador. Estos accesos son utilizados en pantallas donde se prioriza la visión general y donde, de ser necesario, será posible consultar algún aspecto en particular de alguno de los jugadores. Estos accesos son los que se muestran en la Tabla 5.3.

Las acciones que puede realizar aquella persona (docente o administrador) que se encuentra utilizando la herramienta, son el resultado del análisis del Capítulo 4, las cuales ayudarán a solventar las problemáticas encontradas de forma automática y simple. Veremos estas acciones con mayor detenimiento y cómo es su implementación más adelante, cuando se muestren los diferentes ejemplos de uso de la herramienta. Mientras tanto quedarán enumeradas junto con una pequeña descripción. Estas acciones son los que se muestran en la Tabla 5.4.

Por último, en la quinta columna, vemos los íconos que representan los estados de varios jugadores que se encuentran en un mismo sitio. Utiliza una iconografía análoga a la de un jugador solo. Estos íconos ayudarán a visualizar a los jugadores

Icono	Nombre del estado	Descripción
	Seleccionable *(Selectable)*	Un jugador (o grupo de jugadores) se encuentran creados en el sistema pero no se han unido a ningún juego aun.
	Viendo un Tag *(Looking A Tag)*	Al igual que lo mencionando en capítulos previos, este estado representa que un jugador se encuentra viendo un código de dos dimensiones, el cual puede ser o no el que debía ver.
	Viendo una Información *(Viewing Information)*	De la misma manera que lo planteado, este estado representa que un jugador se encuentra una información que le correspondía, información que forma parte del juego.
	Constestando *(Answering)*	El jugador ha recibido una pregunta que forma parte del juego y debe contestarla para poder continuar.
	Caminando *(Walking)*	El jugador se encuentra caminando desde una locación a otra.
	Sesión expirada *(Session Expire)*	Este nuevo estado, indica que el jugador no ha realizado ninguna interacción con el sistema durante cierto tiempo.
	Reiniciar *(Restarting)*	Indica que el jugador ha reiniciado el juego en el cual se encontraba.
	Juego Finalizado *(Finish Play)*	Representa el estado de un jugador al terminar el juego.

Cuadro 5.1: Íconos de estados de los jugadores junto con su nombre y descripción.

Icono	Nombre del Evento	Descripción
	Llegar *(Arrive)*	Este evento sucede al momento que un jugador llega a una escena, tanto sea la correcta o no.
	Redirenccionar *(Redirect)*	Evento que sucede cuando un jugador llega a una escena que no le corresponde y es reorientado hacia la escena correcta.
	Cambio de próximo lugar *(Change Next Place)*	Este evento sucede cuando el jugador terminó de interactuar con la información que se encuentra visualizando en ese momento (por ejemplo una pregunta) y se le modifica el próximo lugar (escena) a la cual ir.
	Comienzo de juego *(Starts A Play)*	Evento que sucede al momento que un jugador se incorpora e inicia un juego.
	Añadir puntos *(Add Points)*	Evento que ocurre al momento que el jugador ha contestado una pregunta de manera correcta.
	Jugador recuperado *(Player Restored)*	Evento que se genera luego de haber realizado una recuperación del jugador desde la base de datos.

Cuadro 5.2: Íconos de los eventos del jugador junto con su nombre y descripción.

Icono	Nombre del Acceso	Descripción
	Visión de los eventos de un jugador *(View events for player)*	El acceso muestra el listado de todos los eventos que fue realizando un jugador en particular dentro del juego en el cual se encuentra jugando.
	Ver el último mapa del jugador *(View last map for player)*	Este acceso mostrará el último mapa que se le entregó al jugador. Esta información será relevante para saber cómo es el comportamiento del jugador frente a los mapas y si es capaz de ubicarse y llegar al destino correcto.

Cuadro 5.3: Íconos de accesos junto con su nombre y descripción.

dentro de los ambientes de juego. En la Tabla 5.5 se ve con mayor detalle cada uno de los íconos.

5.2.2. Juegos

La Figura 5.13 muestra la sección de Juegos (*Plays*). En esta sección se visualiza la estructura de los juegos, es decir la información que forma parte de un juego en particular. Para nuestra implementación de los juegos se ha agregado una introducción dentro de la historia. De esta forma, se podrá poner en contexto a los alumnos antes de ingresar al juego. Luego vemos como en la dentro de esta sección se encuentran todas las informaciones y preguntas que conforman el juego. En la figura se puede ver que tenemos la misma pregunta de ejemplo que vimos en capítulos anteriores (por ejemplo la Figura 2.2). Dentro de la pregunta se pueden ver todas las posibles respuestas que se han cargado. Se puede ver en la imagen que se remarcó cual es el juego seleccionado.

5.2.3. Ambientes de juego

La sección de Ambientes de Juego (*Plays in action*) es visualizada en la Figura 5.14 y la Figura 5.15. Aquí se tienen todas las puestas en escena de los juegos que se encuentran cargados en el sistema. Para cada ambiente de juego, se tiene

Icono	Nombre	Descripción
	Agregar un nuevo jugador *(Add new player)*	Para facilitar el uso del mismo, se crean jugadores dentro del sistema los cuales representarán a los alumnos (o grupos de alumnos) que jugarán.
	Finalizar un jugador *(Finish player)*	Acción que realiza el administrador del sistema para que un jugador finalice (en cualquier momento) el juego en el que se encuentra.
	Reiniciar a un jugador *(Restart player)*	Acción que genera que el jugador deba reiniciar el juego en el que se encuentra. Esta acción no elimina el historial de eventos del jugador.
	Restaurar a un jugador *(Resotore player)*	Acción que restaura al jugador.
	Eliminar a un jugador *(Remove player)*	Elimina a un jugador del sistema.
	Agregar Administrador *(Add Administrator)*	Agrega un nuevo administrador al sistema.
	Eliminar Administrador *(Remove Administrator)*	Elimina un administrador del sistema.

Cuadro 5.4: Íconos de acciones del jugador junto con su nombre y descripción.

Icono	Nombre	Descripción
	Grupo de jugadores viendo un código 2D	Grupo de jugadores que se encuentran viendo el mismo código de 2 dimensiones.
	Grupo de jugadores viendo información	Grupo de jugadores que se encuentran en el mismo lugar viendo información. No necesariamente la misma.
	Grupo de jugadores contestando una pregunta	Grupo de jugadores que se encuentran en el mismo lugar contestando una pregunta. No necesariamente la misma.
	Grupo de jugadores caminando	Grupo de jugadores que se encuentran caminando. No necesariamente con el mismo destino.
	Grupo de jugadores sin actividad	Grupo de jugadores situados en un mismo lugar que no han tenido interacción con el Sistema por cierto tiempo.
	Grupo de jugadores reiniciando el juego	Grupo de jugadores que se encuentran en el mismo lugar reiniciando el juego.
	Grupo de jugadores finalizando el juego	Grupo de jugadores situados en un mismo lugar que no han tenido interacción con el Sistema por cierto tiempo.
	Grupo de jugadores con diferentes estados	Grupo de jugadores que se encuentran en el mismo sitio y que poseen estados diferentes.

Cuadro 5.5: Íconos de estados de grupos de jugadores junto con su nombre y descripción.

Figura 5.13: Información referente a un juego particular.

la siguiente información:

- Lista de los jugadores que se encuentran jugando.
- Un mapa que muestra la distribución de todas las escenas que involucra el juego y la ubicación de los jugadores.
- Relación entre los códigos y las preguntas asociadas a cada uno.
- Lista de los jugadores que hayan finalizado el juego.

Debemos destacar que en las imágenes Figura 5.14 y la Figura 5.15 se han remarcado dos elementos. En la Figura 5.14, se tiene remarcado un código de barras 2D en el lugar donde está posicionado el código de barras 2D y por encima de él la leyenda *G12*. Esa leyenda indica el identificador del código. Se puede observar en la Figura 5.15 que se tiene remarcado el mismo código. De esta forma el docente puede ver el contenido que hay asociado a cada código de barra 2D del juego, y ver a su vez la disposición de los mismos en el mapa.

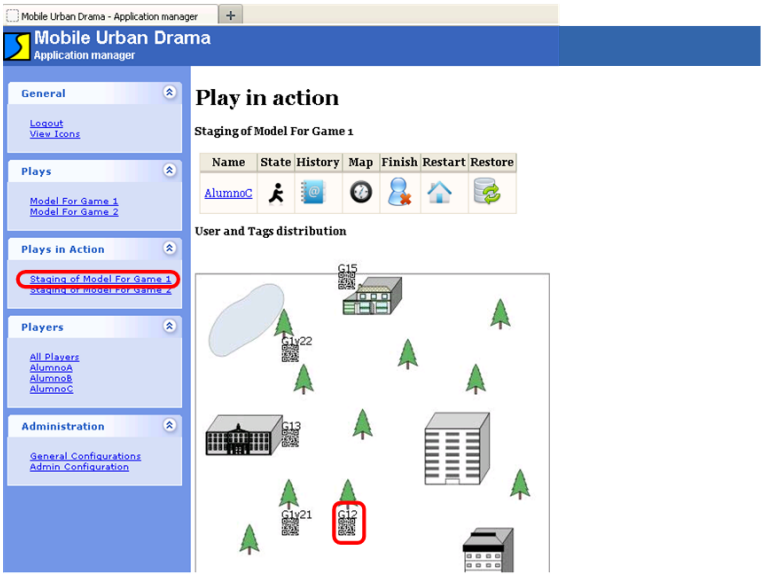

Figura 5.14: Pantalla de un ambiente de juego.

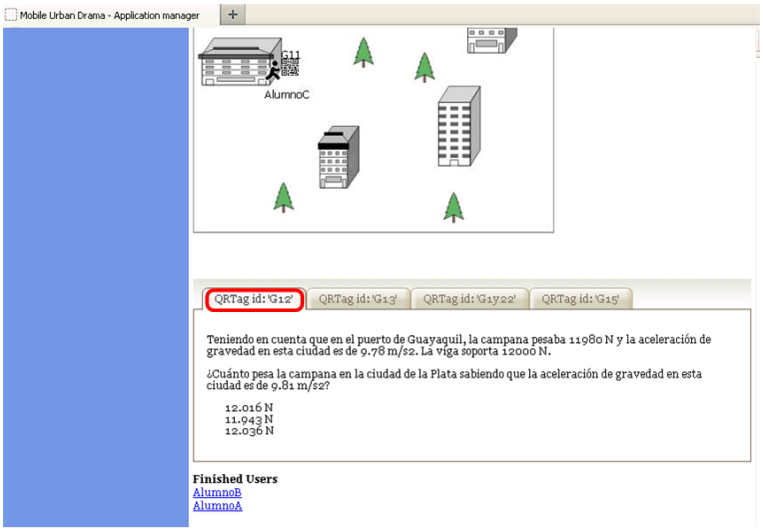

Figura 5.15: Pantalla de un ambiente de juego (continuación).

5.2.4. Jugadores

Veamos que en la sección de Todos los jugadores (*All Players*), como muestra la Figura 5.16, tenemos una fila por cada jugador que existe en el sistema. Dentro de la fila podemos observar que se tiene información sobre el estado del jugador, en qué ambiente de juego se encuentra, ver el historial de eventos y el último mapa. A su vez, el docente (o administrador) puede finalizar el juego actual, reiniciarlo, recuperar a un estado consistente al jugador y por último removerlo

del sistema. También se puede observar más información específica del jugador como en qué escena se encuentra y cuál será la próxima a la cual deberá ir. Podemos ver que en esta sección, se puede tener una visión general de todos los jugadores e inspeccionar cuando se crea conveniente a alguno en particular, ya sea observando desde la misma pantalla la información o accediendo a la pantalla específica del jugador observado.

Por último, podemos observar que se encuentra la opción de crear jugadores, la cual se encuentra remarcada en la figura. El proceso de creación de jugadores se verá luego mediante un ejemplo específico en el Capítulo 6.

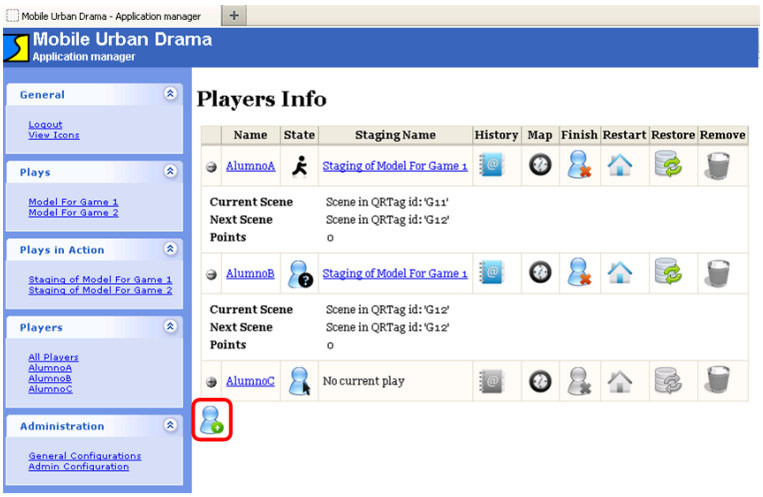

Figura 5.16: Pantalla de información de todos los jugadores.

En la sección de Jugadores (*Players*) se pueden ver todos los jugadores o a cada uno en particular. La Figura 5.17 muestra la pantalla de un jugador particular, en ella podemos ver que la pantalla se divide en tres secciones. Por un lado (arriba a la izquierda), se tiene información referente al estado actual del jugador. Esto es en qué puesta en escena se encuentra jugando (si es que se encuentra en alguna), cual es su estado, en qué escena se encuentra, cuál será la próxima y cuantos puntos tiene el jugador, producto de haber contestado correctamente las preguntas. Además se observa que se encuentran tres acciones que puede ejecutar el administrador. Estas son: finalizar el juego actual, reiniciar el juego actual y recuperar el último estado consistente del jugador. Por debajo se observa el mapa de la puesta en escena en la cual el jugador se encuentra jugando. Este mapa se irá actualizando automáticamente con la ubicación y el estado actual del jugador. Del lado derecho de la pantalla, se tiene el historial de los eventos que el jugador ha ido realizando. Al igual que el resto de los componentes, este historial se irá actualizando en tiempo real con las diferentes acciones que realice el jugador.

Por último, en la Figura 5.17 se encuentra remarcado el link *Login Generator*.

Este link sirve para que en caso que algún jugador tenga que ingresar nuevamente al juego y un docente se encuentra cerca, el docente le podrá generar el código de barras 2D para que pueda realizar el ingreso al juego, sin la necesidad que el jugador se deba trasladar físicamente nuevamente al inicio del juego.

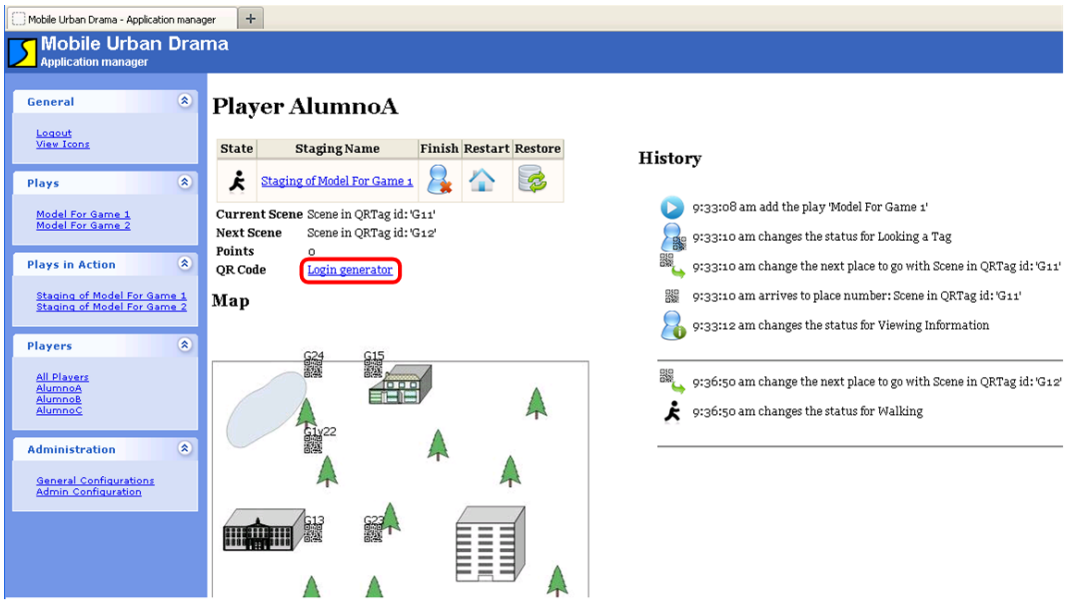

Figura 5.17: Pantalla de visualización de un jugador.

5.2.5. Administración

En la Figura 5.18 se muestra el contenido de la sección de administración del sistema. En ella se puede ver que contamos con la posibilidad de ver y editar el lugar donde se encuentran los archivos persistidos del sistema. Vemos que se encuentran 3 botones diferentes. El primero (*apply*) posee la funcionalidad de aplicar el cambio de directorio donde se persisten los datos. El segundo (*load*), tiene la funcionalidad de cargar toda la estructura de los juegos, puestas en escena y jugadores desde otra posición. Es de notar que por cada juego que se realice será posible almacenarlo en archivos diferentes. El último botón (*Load Example*) carga un ejemplo de dos juegos (*Model for Game 1*y *Model for Game 2*), dos puestas en escena (*Staging for Model for Game 1* y *Staging for Model for Game 2*) y tres jugadores (*AlumnoA*, *AlumnoB* y *AlumnoC*). En particular, uno de los juegos (*Model for Game 1*), junto con su puesta en escena, es una implementación del ejemplo visto en el Capítulo 3.

Cabe destacar que esta sección de la herramienta debe ser utilizada con mucho cuidado, es decir que el docente que ingrese al sistema y realice algún cambio en

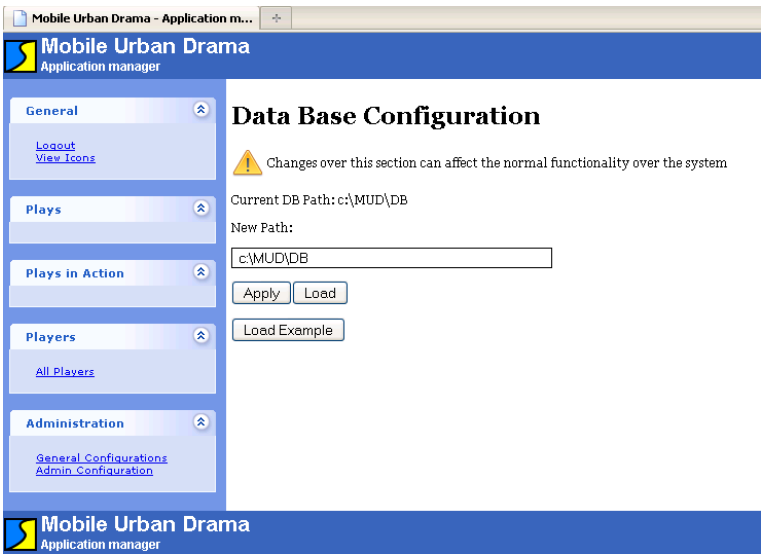

Figura 5.18: Configuración de la herramienta.

la configuración deberá ser conciente en que pueden perderse datos.

La sección de configuración de administradores (*Admin Configuration*) contiene la funcionalidad para crear, editar y eliminar administradores a la herramienta. La herramienta brinda la posibilidad de que distintas personas (docentes) realicen el monitoreo o la administración de la herramienta al mismo tiempo. La Figura 5.19 muestra la pantalla en cuestión.

Figura 5.19: Pantalla de administración de la herramienta.

A lo largo de este capítulo, se ha mostrado una herramienta implementada para monitorear a alumnos que participan en Juegos Educativos Móviles basados en posicionamiento, y que su vez cuenta con capacidades de recuperación ante fallas. Se explicó qué elementos deben colaborar para poder cumplir con los requi-

sitos de monitoreo y recuperación necesarios. Se mostraron las diferentes vistas de la herramienta y qué información y funcionalidad tiene cada una. Quedará por mostrar las ventajas de tener esta herramienta con un ejemplo concreto y con ello mostrar cómo pude ser de gran ayuda tener esta herramienta para integrar los Juegos Educativos Móviles como una actividad más.

Capítulo 6

Ejemplo de uso

En este capítulo veremos cómo utilizar la herramienta presentada en el Capítulo 5 mediante un ejemplo. Este ejemplo se basará en el mismo juego y situaciones presentadas en el Capítulo 2. Para ello, asumimos que ya se poseen las estructuras del juego y sus ambientes de juego ya creados.

Para facilitar, el juego propuesto no presenta la característica de que el jugador pueda elegir hacia que próxima escena dirigirse. El juego de ejemplo es por lo tanto lineal, pero nos es suficiente para mostrar las capacidades de la herramienta y su funcionalidad. También debemos aclarar, que los alumnos son representados (como ya de ha dicho a lo largo de este trabajo) por un jugador digital. Como se dijo, es posible que varios alumnos conformen grupos para jugar compartiendo el mismo dispositivo, haciendo que la experiencia sea grupal. Para este tipo de modalidad, el Sistema utilizará de igual manera un solo jugador o grupo de alumnos ya que el foco esta puesto en el dispositivo que es único por grupo.

Para mostrar el comportamiento de los jugadores a lo largo del juego se utilizarán simuladores de Browsers Mobile, uno diferente por cada jugador. Estos son Opera Mobile [17] para el *Alumno A*, el Opera Mini [16] para el *Alumno B* y por último el Safari [22] para el *Alumno C*.

Cabe aclarar, que el mapa de ejemplo implementado no es exactamente igual al planteado en los capítulos anteriores. Se ha modificado la distribución de los códigos de 2D, y el escenario en dónde se realiza es diferente, ya que se trata de un ejemplo concreto y no de un bosquejo como el presentado en los capítulos

anteriores[1]. Sin embargo, la estructura de las preguntas (es decir su contenido) y la esencia es la misma.

6.1. Creación de Jugadores

Para comenzar con el ejemplo, primero se deben crear todos los jugadores que sean necesarios. Veremos cómo es la secuencia para crear uno de los jugadores y qué pantallas están involucradas.

Dentro de la sección de la herramienta *All Players* está la opción de *Add new Player* mediante un icono. Haciendo click sobre él, aparecerá la pantalla que muestra la Figura 6.1. El formulario presenta los datos que son necesarios para poder crear un nuevo jugador, estos son el nombre y sobre qué ambiente de juego podrá jugar. Esta forma de creación permite que el jugador se encuentre listo para usarse y facilita la interacción del alumno con el Sistema. De esta manera, el alumno utilizará un jugador específico y cuando lea el primer código de 2D ya estará jugando y será posible entregarle información sobre el juego. De otra manera, debería ser el alumno quien cree de forma dinámica al jugador y buscar dentro del Sistema los diferentes ambientes disponibles y seleccionar el correcto al cual ingresar. Esto demanda mayor conocimiento del alumno sobre el Sistema y puede generar más inconvenientes a la hora de usarlo. Por estas razones hemos reducido lo mayor posible la acción de ingreso al juego.

Como muestra la Figura 6.1, estamos creando al *AlumnoA*, el cual desarrollará su actividad dentro de *Staging of Model For Game 1*. La creación del resto de los jugadores es análoga y carece dificultad por lo cual no se mostrará.

6.2. Monitoreo en tiempo real de los Jugadores

Veremos ahora cómo se comporta la herramienta cuando se realiza el seguimiento de los 3 jugadores que se vieron en el ejemplo del Capítulo 2. Haremos primeramente el seguimiento al Alumno A, viendo cómo funciona la herramienta. Luego veremos el mismo ejemplo pero con los 3 jugadores involucrados, de la misma manera que en el ejemplo.

[1]Shapefile con recorrido como soporte de recorridos establecidos.

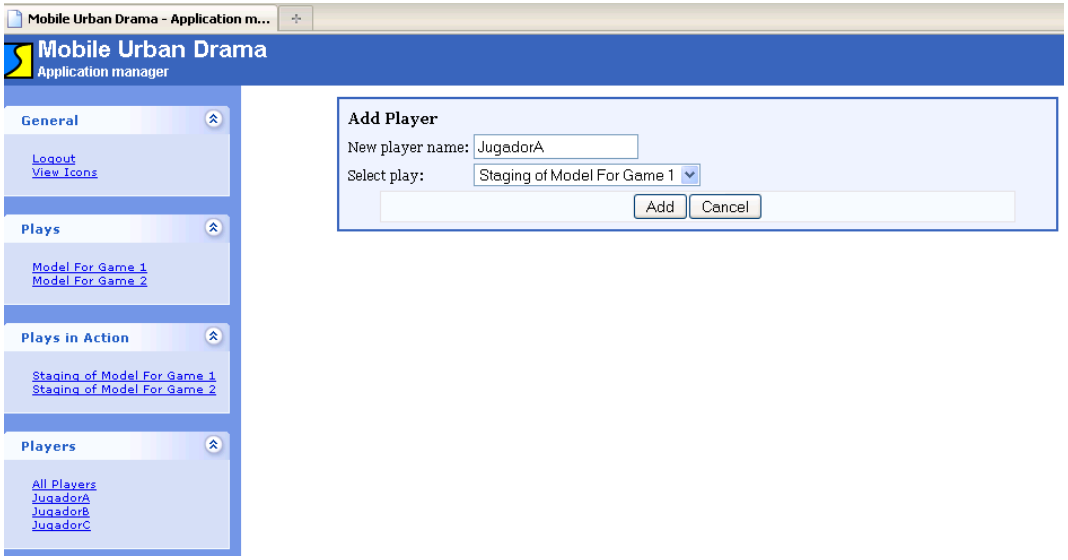

Figura 6.1: Creación del *Alumno A*.

6.3. Monitoreo de un jugador

Primero el alumno debe ingresar al juego, para ello debe leer el primer código de barras de 2D el cual le permitirá comenzar a jugar. La Figura 6.2 nos muestra el resultado de leer el primer código de 2D. Por un lado tenemos la pantalla del docente encargado de realizar el seguimiento, y por el otro, la información que recibió el *Alumno A*. En este caso, el resultado de leer el código fue que el Sistema le muestre una información contextualizando al jugador con el ambiente del juego. Luego que el jugador indique que ya ha visto la información, el Sistema le indicará al jugador hacia dónde debe dirigirse. En cambio, del lado del docente, se puede apreciar que se tiene más actividad involucrada (ingreso al juego, llegada a una escena y ver una información o mapa entre otras). Toda esta actividad que parece irrelevante, nos proporciona los elementos necesarios para luego recuperar al Sistema ante algún fallo, pero esto se verá más adelante.

Una vez que el *Alumno A* se mueve físicamente y llega al lugar indicado por el Sistema, debe leer un nuevo código 2D para avanzar en el juego. Luego de realizarlo podemos observar el resultado en la Figura 6.3, la cual nos muestra por un lado la pregunta que obtuvo el *Alumno A* en su pantalla (la cual debe contestar para continuar) y por el otro, la actualización de la pantalla del docente el cual monitorea al alumno remotamente. Se ha remarcado que el historial de eventos ha crecido, y que el mapa en el cual se visualiza al jugador se ha actualizado. También se ha remarcado que el alumno cambió su ubicación y el icono representativo también se modificó. En este momento, se muestra que se encuentra respondiendo

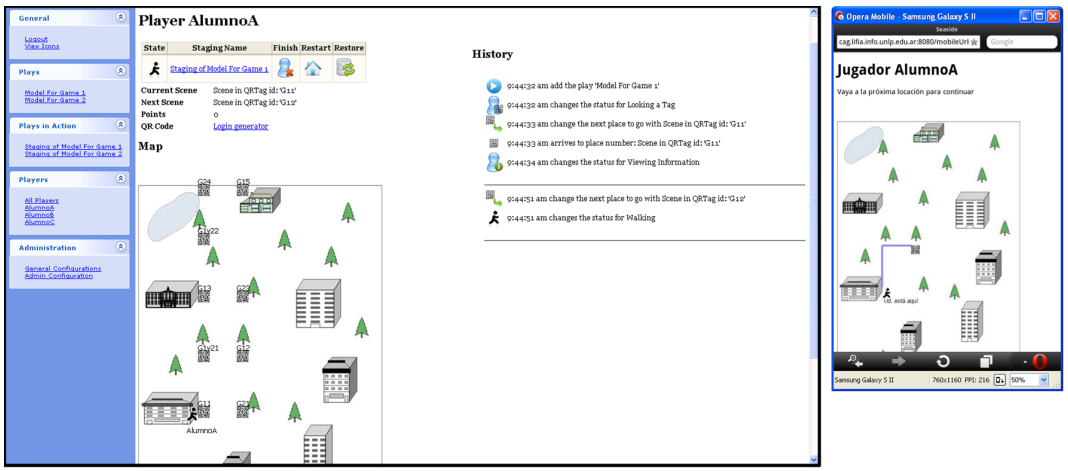

Figura 6.2: Vistas cuando el *Alumno A* ingresa al juego.

una pregunta.

Figura 6.3: *Alumno A* llega a una escena.

La Figura 6.4 nos muestra la situación luego que el *Alumno A* contesta su primer pregunta, vemos que al alumno se le ha brindado un mapa para ir a la próxima escena, mientras que en la pantalla del docente se puede observar que se tiene la pregunta y la respuesta que dio el alumno (junto con evaluación ya realizada) y la actualización del mapa. Se puede ver que el icono ha cambiado por el que representa el estado de caminando *(walking)*.

Continuando con el ejemplo, ahora el *Alumno A* por alguna razón, no llega a la escena que le corresponde (la E3.3) y arriba a una que no le corresponde (la E3.1). La Figura 6.5 nos muestra cómo esta acción se ve reflejada en ambos

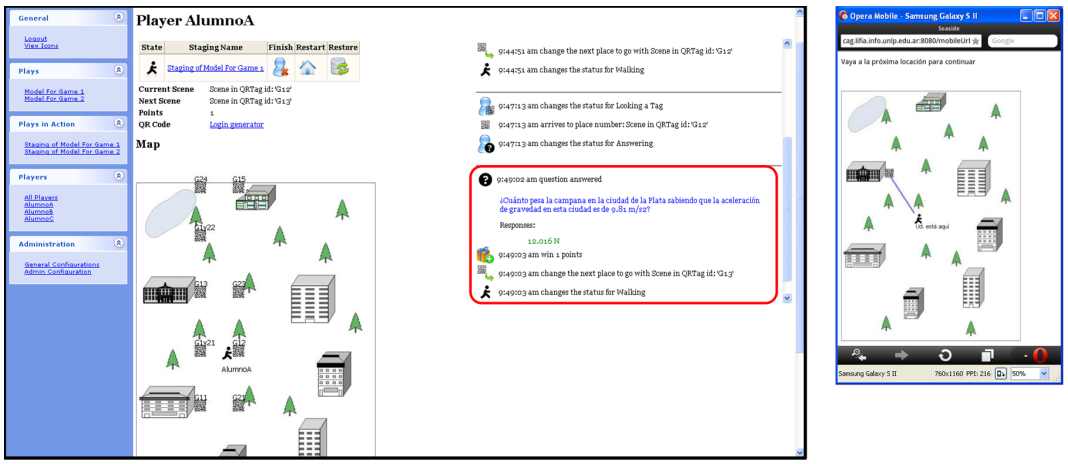

Figura 6.4: *Alumno A* responde una pregunta.

lados, en el docente y en el alumno. El alumno recibe un mapa nuevo indicándole que se ha equivocado, y que debe dirigirse hacia otro lugar para continuar. El docente observa que el alumno ha llegado a otra escena, y puede ver en el mapa que tiene en su vista la nueva ubicación del mismo. A su vez, se puede ver que el historial de eventos ha aumentado y se ha ido modificando acorde a las acciones que realiza el alumno.

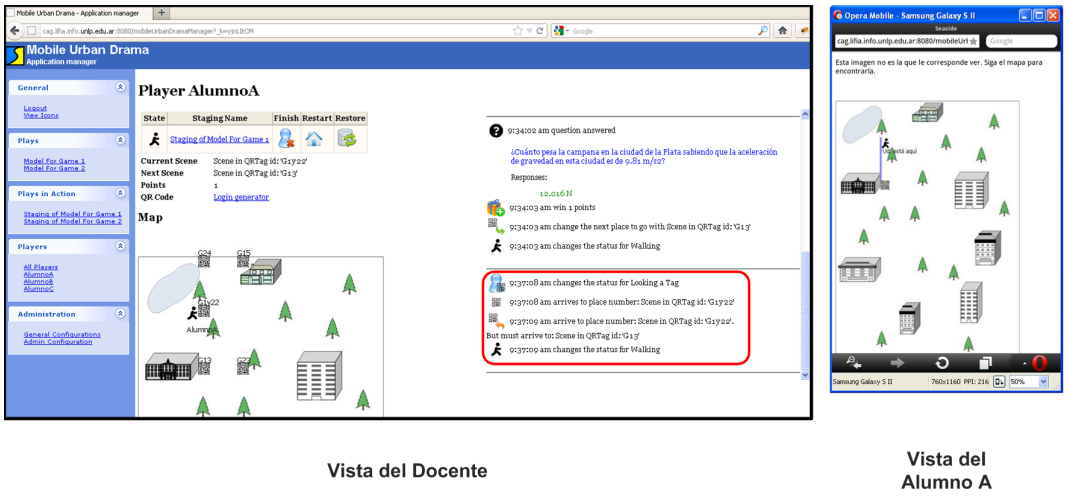

Figura 6.5: El *Alumno A* arriba a una escena incorrecta.

Supongamos que el alumno sigue jugando y contestando, las pantallas de las Figuras 6.3 y 6.4 se repetirán de forma similar. Para finalizar veamos qué sucede cuando el *Alumno A* llega efectivamente a la última escena. Al igual que antes se actualiza la vista del docente, y en este caso, al alumno se le entrega la última pregunta del juego. Una vez contestada, la Figura 6.6 muestra el resultado final

del juego. Por un lado, se tiene la pantalla del *Alumno A* en la cual se muestra el resultado de todas las preguntas que contestó, y en caso de ser necesario, cuál era la respuesta correcta. Por el lado del docente se tiene toda la actividad completa del *Alumno A* dentro de su historial.

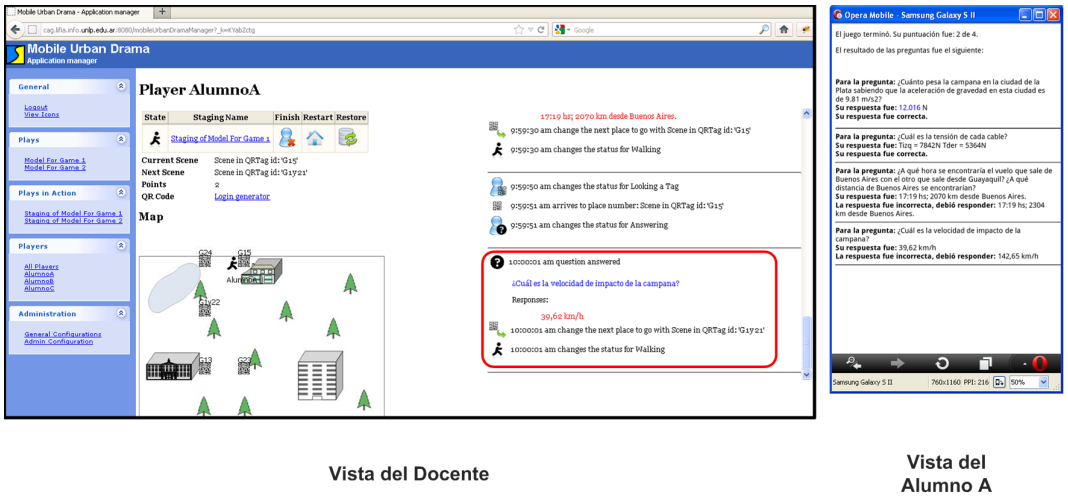

Figura 6.6: El *Alumno A* finaliza el juego.

De esta manera, el docente pudo seguir al *Alumno A* de manera remota sin interponerse en las acciones y decisiones del mismo.

6.4. Monitoreo de tres jugadores

Ahora veamos el caso general donde se tienen varios alumnos jugando un mismo juego. Al igual que en el ejemplo planteado, ahora serán 3 alumnos los cuales se encuentren en el sistema y se disponen a jugar. Todos los jugadores comenzarán desde el principio.

Para realizar el seguimiento de forma más cómoda utilizaremos la sección de la herramienta dedicada a monitorear los ambientes de juego, en este caso en particular, será el *Staging of Model For Game 1*. Una vez que todos los alumnos han leído el primer código 2D, podemos ver en la Figura 6.7 que todos se encuentran en el mismo lugar (la misma escena) y con el estado de 'caminando'. A su vez, dentro de la misma pantalla se tiene un acceso rápido a cada alumno en particular. Se puede ver que todos los alumnos recibieron el mismo mapa de la misma forma que se mostró en el modelo de instancias del modelo propuesto en el Capítulo 3.

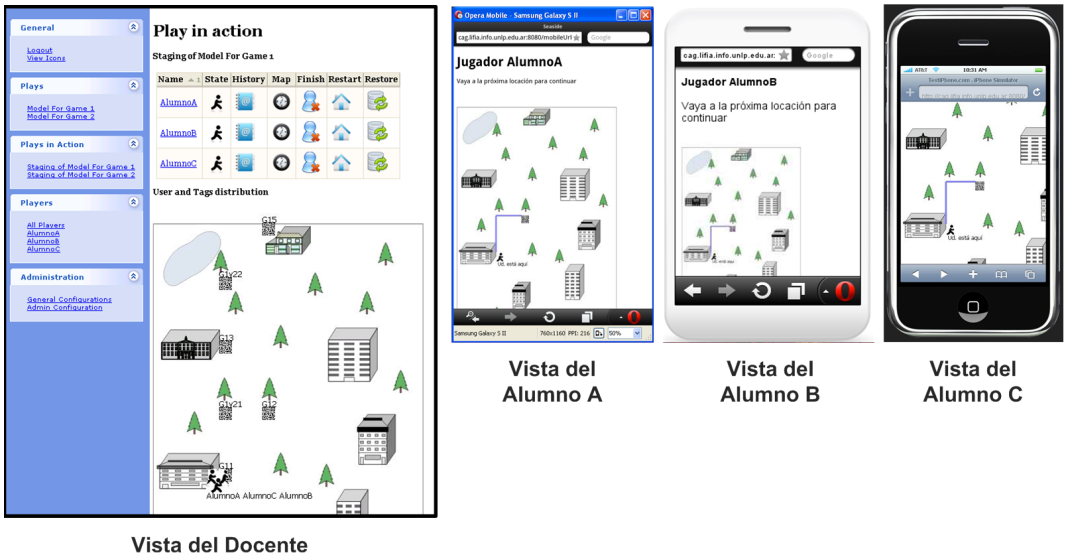

Figura 6.7: Inicio del juego (3 alumnos).

Ahora veremos como cada alumno se desenvuelve de manera distinta y cómo la herramienta es capaz de monitorearlos. A cada alumno se les ha entregado un mapa indicándoles hacia donde dirigirse. Para nuestro ejemplo, dos de los alumnos (el *Alumno A* y el *Alumno B*) han llegado de manera correcta a la escena que les corresponde, mientras que el alumno restante (el *Alumno C*) no ha llegado aún. A los alumnos que han llegado y han leído el código de 2D, se les entregó la pregunta correspondiente. La Figura 6.8 muestra el estado actual del juego. Notamos que como el *Alumno C* aún no ha leído ningún nuevo código, no se tiene más información que se encuentra caminando. Mientras que los otros dos alumnos se encuentran contestando la pregunta.

Ahora veremos como el *Alumno A* ha contestado la pregunta y se ha dirigido a la próxima escena. El *Alumno B* se encuentra en la misma escena que antes y aún no ha respondido la pregunta que se le entregó. Por su parte, el *Alumno C* ha leído un código de 2D y en este caso ha llegado a una escena incorrecta con lo cual se lo debe redireccionar. El estado actual del juego se puede apreciar en la Figura 6.9, donde tenemos la situación descripta anteriormente, y el docente cuenta con las ubicaciones de los alumnos con sus respectivos estados. Si el docente a cargo del monitoreo tuviese la necesidad de obtener mayor datos de alguno de los alumnos en particular, puede hacerlo a través de la barra de iconos que posee cada jugador (ya sea ver el historial o el último mapa generado por el Sistema, entre otros).

Por último veamos cómo la herramienta refleja el hecho que alguno de los alumnos (en este caso el *Alumno A*) termina el juego. Para ello el *Alumno A* ha contestado y se ha traslado físicamente por todas las demás escenas hasta llegar

Figura 6.8: Dos alumnos contestando una pregunta y uno caminando.

a la última, donde ha contestado y se la ha entregado el resultado del juego. El *Alumno C* por su parte ha llegado a la primera escena con pregunta. Mientras que el *Alumno B* ha contestado la pregunta y se encuentra caminando hacia la próxima escena. La Figura 6.10 nos muestra la situación descripta. Podemos ver que el *Alumno A* ya no se encuentra en el mapa brindado al docente ya que no se encuentra jugando.

Figura 6.9: Tres alumnos con distintos estados

Figura 6.10: Un alumno terminó y dos siguen jugando.

Hemos visto como funciona el monitoreo dentro de la herramienta con dos ejemplos, uno centrándonos en un solo jugador y otro con varios jugando al mismo tiempo. Hemos mostrado las pantallas más representativas y que facilitan la comprensión y utilización de la herramienta. Queda claro, que mientras el juego se está desarrollando, el docente que realiza el monitoreo puede cambiar por la vista que le sea más útil para realizar el monitoreo, tanto general como particular de los jugadores.

Por último la Figura 6.11 muestra cómo varios docentes monitorean el mismo juego, cada uno viendo algún aspecto en particular que les interesa sobre los alumnos.

6.5. Recuperación ante fallas

En esta sección abordaremos la funcionalidad de recuperación del sistema ante diferentes fallas. Para ello, se utilizará el juego de ejemplo visto en la sección anterior y se verán distintos tipos de fallas, tanto para uno como para varios jugadores, y cómo hace el Sistema para recuperarse y poder seguir funcionando haciendo posible que los jugadores puedan terminar el juego de manera normal y satisfactoria.

Supongamos que por alguna razón el *Alumno B* decide que desea iniciar nuevamente el juego. Para ello debe avisarle al docente, y éste, utilizando la herramienta

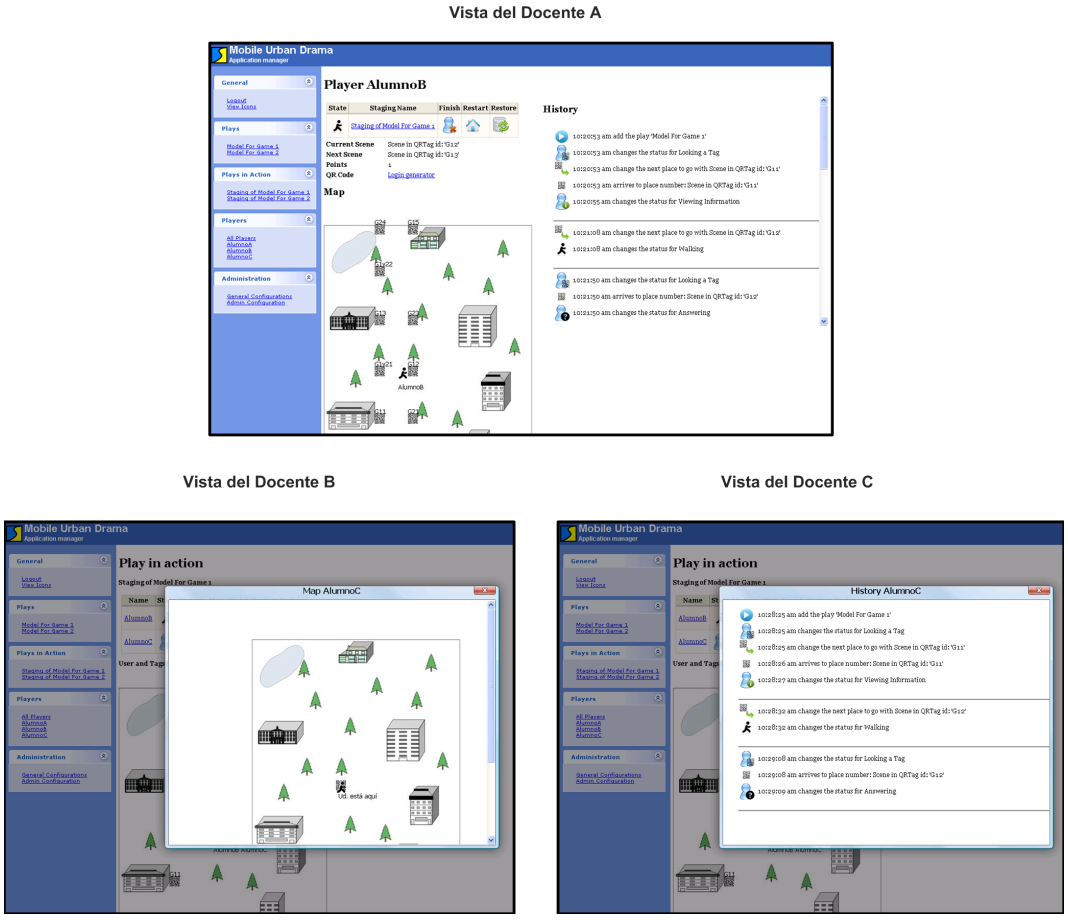

Figura 6.11: Varios docentes observando diferentes datos de los alumnos y del juego.

seleccionará la opción de *Restart* en la sección del jugador correspondiente. Al hacerlo, el *Alumno B* deberá ir a la primera escena y el Sistema le entregará el mapa indicado para ello. Tantos en las secciones del Jugador y del ambiente de juego, los mapas se actualizarán con la nueva ubicación del *Alumno B*. Cabe aclarar que en el historial de eventos del jugador quedará registrado el hecho que ha reiniciado el juego y mantendrá los eventos de previos al reinicio. De esta forma, no se perderán ninguna de las acciones que los jugadores hayan realizado a lo largo de toda la experiencia. La Figura 6.12 muestra este ejemplo.

Veamos ahora qué sucede si los alumnos deben retirarse, y se debe continuar con el juego en otro momento. Para ello, no es necesario realizar ninguna operación sobre el Sistema. Dado que el mismo irá persistiendo los eventos de todos los alumnos constantemente, cuando sea posible que los alumnos puedan continuar y terminar con el juego no será necesario hacer nada. Los alumnos deberán leer los códigos de 2D correspondientes a sus jugadores digitales y ya estarán listos para seguir jugando.

Figura 6.12: El *Alumno B* recomenzó el juego.

Supongamos que alguno de los celulares o dispositivos móviles que poseen los alumnos, se quedan sin batería o quedan inutilizados por alguna cuestión de hardware. Como se ha nombrado en el Capítulo 4, la única solución será la de cambiar el dispositivo ya que sin él la experiencia no se podrá completar. Para ello, los alumnos le informarán al docente y éste les entregará un nuevo dispositivo. Ahora, para poder continuar deberán leer nuevamente el primer código de 2D para ingresar al juego. Al hacerlo el jugador recibirá un mapa indicándole hacia dónde se debe dirigir. De esta manera, el alumno ya está listo para continuar con el juego con el nuevo dispositivo. Internamente, las acciones ocurridas en el Sistema fueron que un nuevo dispositivo se conectó, y tomó al alumno que se encontraba jugando. La Figura 6.13 muestra esta situación.

Figura 6.13: El *Alumno B* cambia de dispositivo.

Tengamos en cuenta el caso en que por alguna de las tantas razones explicadas en el Capítulo 4, el jugador queda con una estructura inconsistente y no puede continuar con la experiencia. El docente, podrá notar esto usando la herramienta observando el perfil del jugador. Si observa que en los datos que se muestran, no se corresponden a los esperados o se tiene una falla, podrá recuperar la última instancia consistente del jugador que se encuentre persistida. Para ello, el docente o administrador del Sistema deberá presionar el icono de *Restore Player*. Esto, como se explicó en el Capítulo 5 (Figura 5.7), recuperará todos los eventos consistentes que se tienen persistidos, y se los aplicará uno a uno al jugador involucrado dejándolo listo para continuar el juego. Esta intervención, puede ser transparente a los alumnos, dado que no requiere una intervención explícita de ellos. La Figura 6.14 muestra este ejemplo.

A lo largo de este capítulo, hemos visto como utilizar la herramienta de seguimiento y recuperación, mediante ejemplos concretos. Podemos observar que con el uso de esta herramienta hemos podido sortear las dificultades presentadas en el Capítulo 4, específicamente en las Secciones 6.3, 6.4 y 6.5. Se ha podido apreciar el beneficio de tener una herramienta de estas características, implementada potenciando aun más el uso de este tipo de actividades en los alumnos y logrando un rol activo de los docentes.

Figura 6.14: Recuperación del *Alumno C*.

Capítulo 7

Conclusiones y Trabajo Futuro

En este capítulo se presentan las conclusiones obtenidas luego del análisis y la implementación de la herramienta. Luego se describirán los posibles trabajos a futuro.

7.1. Conclusiones

Se presentó un modelo que muestra las características básicas de los juegos educativos móviles basados en posicionamiento. En estos juegos se hace hincapié en la posición donde se encuentra el jugador, y sobre la actividad que está desarrollando. Para este trabajo, se tomó como interacción entre los jugadores y el juego la formulación de preguntas. Mediante la respuesta de los jugadores el flujo del juego irá cambiando y le brindará a los jugadores nuevos lugares a los cuales ir y preguntas para contestar.

Se profundizó sobre el rol que cumplen los jugadores en estos juegos en particular. Se realizó un análisis sobre las actividades que desarrollan los mismos a lo largo del juego y el impacto que tienen. Se hizo necesario modelar todas estas características para la implementación de los mismos, para poder realizar pruebas.

Mediante la implementación del modelo propuesto, se pudieron descubrir problemáticas específicas sobre estas aplicaciones. Las problemáticas encontradas

fueron de diferentes tipos, tales como tecnológicas (el dispositivo móvil se queda sin batería o queda inutilizable, entre otros) o incorrecta interacción con el juego (los jugadores no leen de forma correcta los códigos de 2 dimensiones propuestos, realizan actividades incorrectas, entre otros).

Estas problemáticas hicieron que fuese necesario implementar una herramienta de monitoreo y recuperación, para que el uso de estos juegos sea viable. Para poder desarrollar estas herramientas, fue necesario observar el modelo propuesto y observar sobre qué elementos era necesario mantener un seguimiento exhaustivo sin modificar el modelo propuesto. De esta manera, se buscó que las herramientas implementadas tuvieran la capacidad de poder monitorear y recuperar diferentes tipos de modelos propuestos mientras, contara con un mecanismo de aviso de cambio.

Las herramientas implementadas pudieron resolver varias de las problemáticas encontradas, tales como: la recuperación ante fallas propias de los dispositivos móviles, la capacidad de recomenzar los juegos en cualquier momento, la capacidad de observar a los jugadores de una manera transparente para ellos y en tiempo real.

Se pudo ver que a lo largo del trabajo se trató de manera indistinta el concepto de alumno con el de jugador, dado que para este tipo de herramientas la división entre los conceptos no es necesaria.

7.2. Trabajo a Futuro

Hemos mostrado en este trabajo una herramienta prototípica que simplifica el trabajo de los docentes, y brinda información importante para que ellos puedan utilizarla de la manera que crean conveniente para evaluar y enseñar a los alumnos. Pero creemos que esta herramienta es un primer paso para la inclusión de este tipo de aplicaciones. Debemos destacar, que es necesario continuar con este trabajo desde diferentes perspectivas, dado que el objetivo de este trabajo fue el generar una primera herramienta prototípica y observar el comportamiento de los docentes y alumnos con ella. Se pueden distinguir diferentes áreas en las cuales existe trabajo por hacer. Entre ellas distinguimos:

- Herramienta de monitoreo. Creemos que se puede agregar comportamiento y diferentes aspectos a la herramienta para enriquecerla aún más.
    - El aspecto visual de la herramienta necesita ser mejorado, para que sea

aún más fácil de comprender por los docentes y que pueda brindarles la información de manera más clara y concisa.

- La posibilidad de tener un chat entre los alumnos y los docentes, para que si realizan el monitoreo de forma remota, puedan interactuar entre ellos sin la necesidad de trasladarse físicamente.
- Poder tener una vía de comunicación entre docentes y tutores (de tipo chat) sería interesante. Con ello se podría ayudar a los docentes a que la comunicación entre ellos sea fluida, sin importar la distancia en la que se encuentren con los grupos de alumnos.
- Durante las pruebas realizadas, hemos descubierto que varios docentes que realizaban el seguimiento acompañando a los alumnos, veían cierto comportamiento especial que les llamaba la atención y debían recordarlo por fuera de la herramienta. Creemos que una mejora a la herramienta, sería poder brindarles a los docentes la capacidad de guardar notas de los alumnos que se encuentran jugando. Además de modificar la vista del docente, para éstos docentes que ya se encuentran con los alumnos, ver qué tipo de información se podría mostrar para que le sea útil, por ejemplo será necesario que los mapas que vean estén coordinados con el de los alumnos a los cuales siguen. Relacionado con esto ver de qué forma estas notas de comportamiento se pueden vincular entre las preguntas, y entre otros docentes para obtener datos generales sobre el juego y sobre todos los jugadores.
- Otro aspecto a agregar a la herramienta es el muestreo de datos estadísticos de forma clara y concisa. Es decir, poder mesurar de una forma clara, cuantos grupos han contestado bien una pregunta específica y en cuanto tiempo, cuanto a ha sido el tiempo promedio para la finalización de un juego en particular, cuantas veces los grupos se han equivocado al leer un código 2D y han llegado a otra escena, por nombrar algunos. Los datos se encuentran dentro del Sistema, ya que de alguna u otra manera se encuentran modelados. Es cuestión de recorrer el modelo propuesto y poder visualizar estos datos. Poder persistir estos datos y todos los comportamientos de los juegos sería de gran utilidad. Tener los datos históricos de los alumnos, ayudaría a comprender mejor el progreso obtenido a lo largo del tiempo.

- Herramienta de recuperación. Si bien la herramienta de recuperación cumple con su objetivo, creemos que es necesario realizar más trabajo sobre ella.

  - Por ejemplo, dar la posibilidad de restaurar o recuperar tanto un jugador como a un juego en algún punto en particular. Dándole así una herramienta más versátil y que permita a quien la use mayor libertad de acción.

- Enriquecer los juegos de diferentes modos. Lo expuesto en este trabajo mostró juegos simples y sin mucha complejidad, pero es posible crear juegos más complejos. Con este objetivo, creemos que pueden ser importantes tener diferentes tipos de mejoras al modelo de los juegos planteados y con ello a las herramientas descriptas.

  - Probar la herramienta con grupos masivos de gente. Es decir, con una gran cantidad de alumnos y varios docentes observando indistintamente el comportamiento de los mismos. Por ejemplo utilizar varios cursos de alumnos de distintas edades que se encuentren jugando varios juegos. Con estas pruebas, podremos detectar la eficiencia y el tiempo de respuesta de la herramienta y verificar si es el óptimo o no.
  - Poder hacer que los alumnos puedan jugar varios juegos y que puedan decidir a qué juego jugar en un lapso de tiempo específico. Es decir, poder soportar el ingreso y egreso de jugadores dentro los juegos de forma más dinámica.
  - Diseñar juegos para que los alumnos deban conformar grupos o relacionarse de forma dinámica durante el juego. Es decir, que durante un mismo juego, los alumnos deban ir asociándose con otros para cumplir ciertos objetivos del juego o para compartir información. Para ello será necesario adentrarse en la estructura de los juegos y revisar el modelo provisto en este trabajo para modelar a los alumnos y sus acciones dentro del juego. Además de modificar la herramienta de monitoreo para que este nuevo tipo de interacción se pueda observar de forma simple para el docente.
  - Agregar comportamiento a la hora de responder las preguntas. Por ejemplo, si en cierto tiempo no se ha respondido, otorgarle al grupo una ayuda que podría ser más información o quitar una de las opciones incorrectas de las respuestas. También se podría premiar a aquellos grupos que tengan un buen desenvolvimiento dentro del juego modificando en forma dinámica la cantidad de opciones a responder en las preguntas.
  - Durante este trabajo los juegos creados fueron simples, pero no por ello difíciles de construir. Muchas de las características de este tipo de juegos fueron hechas de forma *ad hoc*, por ejemplo los mapas de distribución de las escenas y de sus correspondencias dentro de los mismos. Para poder incluir este tipo de aplicaciones en las escuelas es necesario poder contar con una herramienta que pueda utilizar un docente (o alguna persona sin mayores conocimientos en informática) para generar el contenido que crea necesario. Para ello será conveniente realizar un análisis más profundo sobre los juegos y su estructura que el presentado en este trabajo. Será a su vez importante que esta nue-

va herramienta pueda combinarse con la presentada en este trabajo, dándole a los docentes un conjunto de herramientas consolidadas para que su manejo sea simple y fácil.

- Es necesario realizar un estudio sobre los perfiles de las personas que vayan a utilizar la herramienta, y ver si es necesario o no crear diferentes vistas dependiendo del perfil de cada persona, por ejemplo docentes, tutores y administradores. Creemos que es importante que aquella persona que vaya a utilizar esta herramienta pueda configurarla de forma tal que le sea simple y fácil de utilizar, de forma tal que le ayude a resolver diferentes inconvenientes y no generarles nuevos.
- Además, sería importante poder reutilizar los contenidos generados por los docentes (los juegos, las preguntas y las historias como hilo conductor). Es decir que de alguna manera un docente que ha creado un juego pueda reutilizar las preguntas que ha formulado otro docente en otro juego. Creemos firmemente que este tipo colaboración, ayudará a que las aplicaciones educativas móviles basadas en posicionamiento puedan llegar a ser tomadas en cuenta como herramienta para el aprendizaje. Para ello utilizar algún tipo de tecnología como repositorios facilitaría esta característica que no se encuentra presente en ninguna de las aplicaciones vistas. Compartir el trabajo realizado ayudará rápidamente a la generación de juegos ricos en contenido y entretenidos para los alumnos. También creemos que la infraestructura que poseen los repositorios aumenta la cantidad de información sobre los juegos, es decir, que se puede votar por los juegos, discutir sobre si para evaluar un tema en particular un juego en específico es mejor que otro, poder buscar juegos afines, por nombrar algunos.

Con esto hemos terminado este trabajo, podemos apreciar que el trabajo a futuro es mucho y que existen diferentes líneas de trabajo para seguir. Creemos que a su vez que será importante integrar otras áreas de trabajo como la pedagógica para continuar con el estudio de estos juegos y poder mejorar su dinámica aún más.

Bibliografía

[1] S. E. G. Alejandra B. Lliteras, Cecilia Challiol. Juegos Educativos Móviles Basados en Posicionamiento: Una Guía para su Conceptualización. *Proceding of Argentine Symposium on Software Engineering (ASSE) 2012 - 41 JAIIO*, pages 164–175, 2012.

[2] BeeTagg. BeeTagg. `http://www.beetagg.com/en/`, 2012. [Online; accedido por última vez en Diciembre del 2012].

[3] S. Benford, D. Rowland, M. Flintham, A. Drozd, R. Hull, J. Morrison, and K. Facer. Life on the Edge : Supporting Collaboration in Location-Based Experiences. *Interface*, pages 721–730, 2005.

[4] S. Björk, J. Falk, R. Hansson, and P. Ljungstrand. Pirates! Using the Physical World as a Game Board. *Design*, 2001.

[5] N. Braun. Storytelling in collaborative augmented reality environments. *Proceedings of the 11th International Conference in*, 2003.

[6] V. Bykov. Introducing Announcements. `http://www.cincomsmalltalk.com/userblogs/vbykov/blogView?showComments=true\&entry=3310034894`, 2012. [Online; accedido por última vez en Diciembre del 2012].

[7] Cincom. Cincom VisualWorks - Cincom Smalltalk. `http://www.cincomsmalltalk.com/main/products/visualworks/overview/`. [Online; accedido por última vez en Diciembre del 2012].

[8] D. Crane and P. McCarthy. Comet and reverse ajax. 2008.

[9] E. Gamma, R. Helm, and R. Johnson. Design patterns: elements of reusable object-oriented software. 1995.

[10] F. A. Hansen and K. J. Kortbek. Mobile Urban Drama for Multimedia-Based Out-of-School Learning. *Framework*, 2010.

[11] F. A. Hansen, K. J. Kortbek, and K. Gronbek. Mobile Urban Drama - Setting the Stage with Location Based Technologies. *Technology*, pages 20–31, 2008.

[12] J. Huizenga, W. Admiraal, and S. Akkerman. Learning history by playing a mobile city game. *game-based learning*, (2005), 2007.

[13] J. Kjeldskov. Augmenting the City with fiction: fictional requirements for mobile guides. *Mobile Interaction with the Real World*, pages 1–6, 2007.

[14] M. Li. Geo-Gaming: The Mobile Monopoly Experience. *Information Systems*, (May):2006–2009, 2008.

[15] A. Lieberoth. Can Autobiographical Memories Create Better Learning? The Case of a Scary Game. *pure.au.dk*, pages 1–9.

[16] O. Mini. Opera Mini. `http://demo.opera-mini.net/public/index.html`, 2012. [Online; accedido por última vez en Diciembre del 2012].

[17] O. Mobile. Opera Mobile. `http://www.opera.com/developer/tools/mobile/`, 2012. [Online; accedido por última vez en Diciembre del 2012].

[18] MPP. Metro Paris Phone. `http://www.metroparisiphone.com`, 2012. [Online; accedido por última vez en Diciembre del 2012].

[19] QuickMark. QuickMark an easy-to-use barcode scanner. `http://www.quickmark.com.tw/En/basic/index.asp`, 2012. [Online; accedido por última vez en Diciembre del 2012].

[20] J. Raessens. Playing History : Reflections on Mobile and Location-Based Learning. *Society*, 2007.

[21] S. Robles and L. Fernández. *Meteoroid : Un MVC real para la Web*. PhD thesis, 2010.

[22] Safari. Safari. `http://www.testiphone.com`, 2012. [Online; accedido por última vez en Diciembre del 2012].

[23] ScanLife. ScanLife. `http://www.scanlife.com/en/`, 2012. [Online; accedido por última vez en Diciembre del 2012].

[24] Seaside. Seaside. `http://www.seaside.st`, 2012. [Online; accedido por última vez en Diciembre del 2012].

[25] S. Seol, A. Sharp, P. Kim, S. Empowerment, and P. Alto. Stanford Mobile Inquiry-based Learning Environment (SMILE): using mobile phones to promote student inquires in the elementary classroom Stanford Mobile Inquiry-based. 2011.

[26] K. G. Stanley, I. Livingston, A. Bandurka, R. Kapiszka, and R. L. Mandryk. PiNiZoRo : A GPS-based Exercise Game for Families. pages 243–246, 2010.

[27] M. Wijers, V. Jonker, and P. Drijvers. MobileMath: exploring mathematics outside the classroom. *Zdm*, 42(7):789–799, Sept. 2010.