UNIVERSIDAD POLITÉCNICA DE MADRID

FACULTAD DE INFORMÁTICA

TRABAJO FIN DE CARRERA

APLICACIÓN DEL MÉTODO DE LARMAN A LA CONSTRUCCIÓN DE ENTORNOS VIRTUALES

AUTOR: **GONZALO MÉNDEZ POZO**

TUTORAS: **ANGÉLICA DE ANTONIO JIMÉNEZ**

MARÍA ISABEL SÁNCHEZ SEGURA

Le dedico este libro a mis padres y a mi hermano, que después de 24 años de esfuerzo han conseguido sacar algo bueno de mí.

Gracias a Angélica de Antonio por todas las oportunidades que me ha dado, a Maribel por estar siempre cerca cuando me ha hecho falta, a Xavi, Óscar, Chema, Ricardo, Gus y toda la gente del Laboratorio Decoroso Crespo, por hacer estos tres años tan entretenidos, y a todos mis amigos de dentro y fuera de la facultad por seguir aguantándome después de tanto tiempo.

ÍNDICE

ENTORNOS VIRTUALES HABITADOS

1. INTRODUCCIÓN

Con el crecimiento de las redes de ordenadores y especialmente de la red Internet, los usuarios de las mismas han encontrado un nuevo atractivo en aplicaciones como *Chats*, *MUDs*, *EVs* y, en general, aplicaciones en las cuales la idea subyacente no es la interacción con el sistema, sino con otros usuarios que hacen uso de la misma aplicación u otra similar en distintos lugares del mundo.

El primer MUD *(Multi-User Dungeon)* fue desarrollado por Roy Trubshaw y Richard Bartle en 1978. En sus comienzos, el término MUD hacía referencia a un juego de aventuras concreto, y posteriormente empezó a referirse a todo tipo de entornos textuales a los que la gente se conectaba para participar en juegos de aventura multijugador.

En 1989 Jim Aspnes, en la Carnegie Mellon University, desarrolló lo que se denominó *"TinyMud"*. Se trataba de un nuevo tipo de MUDs en los que el objetivo principal no era el de alcanzar una meta y vencer a los contrincantes, sino el de entablar conversaciones, conocer gente, etc. Fueron llamados *SocialMUDs* [Dourish98].

Al mismo tiempo que evolucionaban los MUDs, la tecnología empezaba a permitir llevar a cabo los primeros experimentos para dotar de una interfaz gráfica a las comunidades virtuales basadas en texto; el resultado de estos experimentos recibió el nombre de *"Entornos Virtuales"* (EV). Fue en 1985 cuando Lucasfilm creó el primer EV gráfico llamado *Habitat*. Se trataba de un entorno bidimensional, donde los usuarios estaban representados por un a*vatar***,** que es la representación gráfica de un usuario dentro del EV. Los usuarios podían manejar su avatar mediante el teclado del ordenador y comunicarse, utilizando el lenguaje escrito, con los otros usuarios conectados.

Desde entonces han ido surgiendo multitud de EVs de este tipo. Algunos han ido introduciendo visualización tridimensional (3D), sonido, capacidad para crear nuevos objetos en el EV, etc., siempre con el ánimo de ir haciéndolos más amigables para los usuarios.

Por otro lado, a la vez que iban surgiendo nuevas actividades a realizar en estos EVs y a medida que iban aumentando las posibles acciones que podían realizar los avatares, también iba creciendo la complejidad en el manejo de los mismos, de manera que cada vez se hacía necesario un mayor número de controles para poder usarlos. Con

el objetivo de simplificar la labor del usuario surge la idea de dotar a los avatares de comportamiento, de manera que cierto tipo de acciones pueda llevarse a cabo sin una orden expresa de quien utilice el EV.

Bodychat [Vilhjálmsson97] es un EV en el que se intenta automatizar el comportamiento de los avatares. Aquí se introduce el concepto de *Intención,* que se puede describir como el conjunto de parámetros de control que se envían desde un cliente, al que está conectado un usuario, al resto de clientes conectados. Dichos parámetros de control servirán para reflejar, en cada uno de los clientes, el comportamiento del avatar que representa al primer cliente.

Además de los mencionados, existen una serie de proyectos y sistemas que se han desarrollado centrándose en la representación de los cuerpos virtuales. El objetivo de estos sistemas es crear avatares cada vez más creíbles, de modo que el usuario real se sienta cada vez más identificado con su avatar.

Como se puede ver, estos sistemas han ido evolucionando desde unas simples aplicaciones en modo texto a complejos desarrollos que incluyen imágenes 3D con alto grado de detalle, sonidos y un grado de autonomía variable, dependiendo del propósito del sistema. Y son todas estas características las que hacen que estos sistemas sean distintos de los sistemas tradicionales, como procesadores de texto o sistemas de información, en los que lo realmente importante es que funcionen bien, que el resultado final sea correcto y sea el esperado. Por el contrario, a un EV, además, hay que dotarlo de credibilidad, es decir, que no sólo es importante que internamente todo funcione bien, sino que hay que enseñarle al usuario todo el proceso a través de unas imágenes y un sonido; lo importante no es sólo que un avatar se ponga triste y el sistema funcione como si el avatar estuviese triste, sino que eso hay que mostrárselo al usuario con el suficiente grado de realismo como para que el usuario se lo pueda creer. En resumen, que no sólo importa el resultado, sino que todo el proceso sea visible, y en esto reside gran parte de la complejidad y del atractivo de este tipo de aplicación.

Un EV puede funcionar como una aplicación tradicional, pero siempre cuenta con la característica añadida de tener que representar lo que el sistema está haciendo para que se pueda interactuar de manera más sencilla en los casos en que sea necesario, ya que uno de los principales usos que se les da es reproducir una situación real para que se pueda interactuar con todos los objetos y avatares que en ella aparecen.

Con el fin de comprobar el estado actual de los entornos virtuales, se ha realizado un pequeño recorrido por los proyectos sobre desarrollos de EVs más importantes en la actualidad, y los datos que se han encontrado son los que se muestran a continuación:

Proyecto COVEN [Coven99]

- Aplicado principalmente a la investigación sobre la construcción de EVs de cualquier tipo.
- Enfocado hacia cualquier tipo de usuario.
- El objetivo principal es estudiar las características de diseño, implementación y utilización de un EV multiusuario a nivel científico, técnico y metodológico, teniendo en cuenta básicamente 4 factores: diseño de elecciones, usabilidad, facilidad en el desarrollo y velocidad de ejecución.
- La plataforma de desarrollo está basada en DIVE. Para los modelos 3D utilizan el modelador AC3D, de la Universidad de Lancaster. La comunicación en red está implementada siguiendo los modelos de DIVE y MASSIVE. Parte del comportamiento de los objetos se ha desarrollado usando dVS.
- Interacción con el sistema usando ratón, teclado y micrófono.
- Posibilidades de Interacción dentro del Sistema:
  - Los usuarios pueden navegar por el EV y pueden interactuar entre sí y con los objetos del EV.
  - La comunicación con otros usuarios se puede hacer en modo texto o bien se pueden comunicar por medio de la voz.
  - Los usuarios están representados por avatares con forma humana, aunque se puede cambiar esta representación a gusto del usuario, utilizando una especie de bolas en su lugar, con un mecanismo que denominan de ‘vistas subjetivas’. Los avatares con forma humana tienen un diseño bastante realista, y están construidos con *metaballs*. El movimiento se ha diseñado usando datos antropométricos, y se ha implementado con técnicas de cinemática inversa.
  - Existen agentes, pero no interactúan demasiado con los usuarios.

- Se hace detección de colisiones.
- El entorno es multiusuario, y no se expone ningún límite en cuanto a número de usuarios conectados.

♣ El proceso que se sigue es el desarrollo y posterior evaluación de tres versiones sucesivas de un prototipo de aplicación. Los dos primeros ciclos han dado lugar a una serie de técnicas de interacción y colaboración mediante el uso de 'vistas subjetivas'. El tercero servirá para desarrollar técnicas que faciliten la escalabilidad de los EVs, sobre todo en términos de facilitar la renderización de EVs de grandes dimensiones. Han diseñado un método para probar la usabilidad del sistema (inspección y observación).

♣ Han contribuido al desarrollo de estándares en dos áreas: Web3D (3D Worlds over the Internet, antiguo VRML) y MPEG4-SNHC (Face Body Animation). También ha aportado datos para el desarrollo de HLA/DIS (high-level architectures / distributed interactive simulation).

Proyecto DIVE [Dive99]

♣ Aplicado al desarrollo de EVs, interfaces y aplicaciones basadas en entornos 3D multiusuario.

♣ Dirigido a cualquier tipo de usuario.

♣ El propósito es construir un EV y dotar a los objetos de comportamiento, por lo que se pueden cargar nuevos objetos, moverlos, rotarlos, añadirles comportamientos.

♣ Utilizan VRML y otros formatos 3D para los objetos y Tcl/Tk para los scripts que definen el comportamiento de los objetos, los cuales se disparan con ciertos eventos el sistema: interacciones entre los usuarios, colisiones, temporizadores, etc.

♣ Se controla con ratón y/o teclado.

♣ Posibilidades de Interacción dentro del Sistema:

- El propósito es construir un EV y dotar a los objetos de comportamiento, por lo que se pueden cargar nuevos objetos, moverlos, rotarlos y añadirles comportamientos.
- Entre usuarios, existe comunicación oral y escrita.
- Hay avatares que representan al usuario, con varios puntos de vista. Se puede cambiar el avatar sobre la marcha.
- Hay agentes.
- Se realiza detección de colisiones.
- Es multiusuario, y se integra con la World Wide Web.

♣ No existe ninguna metodología de desarrollo; todo lo más, se adaptan al estándar existente para VRML.

Proyecto MASSIVE [Massive95]

♣ Es un sistema de teleconferencia y un ejemplo de implementación de interacción espacial.

♣ Está pensado para que lo use cualquier tipo de usuario que haga uso de la teleconferencia.

♣ El objetivo es construir un sistema de teleconferencia que implemente una primera aproximación de un modelo de interacción espacial.

♣ La comunicación en red se ha conseguido programando con RPCs.

♣ Se maneja con teclado, ratón o micrófono.

♣ Posibilidades de Interacción dentro del Sistema:

- Es un entorno sólo para ser visitado, por lo que no hay mucha interacción.
- Interacción muy básica con otros usuarios, a nivel de colisiones y conversación oral, escrita o visual.
- Hay tres formas de utilizarlo: texto, gráficos y audio. Los usuarios con texto y gráficos pueden interactuar entre sí.
- No hay agentes.

- Detección de colisiones usando la técnica de *bounding boxes* (les llaman auras).
- Multiusuario con un límite aproximado de 10 usuarios.

♣ Sin metodología definida.

♣ Continuado en Massive-2, de características similares pero implementado con JAVA. Admite un mayor número de usuarios.

Proyecto AVIARY [Aviary94]

♣ Aviary es un entorno virtual multiusuario genérico; básicamente, es una arquitectura con la cual se puedan construir EVs de distintas características.

♣ Por las características mencionadas anteriormente, está especialmente orientado hacia personas o grupos cuyo objetivo sea la construcción de EVs.

♣ El objetivo principal es la construcción de una aplicación que permita el desarrollo de diferentes mundos, cada uno de los cuales tendrá sus propias características y leyes que definan el comportamiento de los objetos que se encuentran entro de él.

♣ Diseñado para aprovechar características de sistemas multiprocesador o distribuidos. La implementación consiste en la ejecución concurrente de distintos objetos autónomos.

♣ Posibilidades de Interacción dentro del Sistema:

- La interacción depende del tipo de entorno que se construya; varía desde simples entornos para ver lo que hay hasta un alto grado de interacción entre los usuarios.
- La existencia de avatares, al igual que la interacción, depende de los distintos tipos de EVs que se quieran construir.
- Aunque no se menciona explícitamente la existencia de agentes, cada objeto del EV tiene un demonio que controla sus acciones, por lo que puede llegar a comportarse como un agente.

- Se contempla la posibilidad de considerar o no la detección de colisiones, dependiendo de las necesidades. Para realizarla, se utiliza la técnica de *bounding boxes*. Además, cuando existen muchos objetos se divide el volumen del EV en partes, de manera que al mover un objeto sólo se comprueban las colisiones con objetos que estén en la misma zona del EV. Esto lo consiguen con la creación de EDBs (Environment Database).
- Es multiusuario, y se ha construido pensando en la conexión simultánea de un alto número de usuarios.

♣ Se ha realizado la codificación con un lenguaje orientado a objetos.

Proyecto DEVA [Deva99]

♣ Está aplicado a la construcción de EVs en general.

♣ Este sistema se dirige a usuarios que se dediquen a la creación de EVs, especialmente si son de gran tamaño y permiten la conexión de muchos usuarios.

♣ El principal objetivo es crear un sistema de realidad virtual que dé soporte a EVs multiusuario de grandes dimensiones. Esto se hace apoyándose en la experiencia previa de Aviary y Maverik. En concreto, los principales problemas a resolver son la descripción del comportamiento de los objetos, el manejo de múltiples aplicaciones y la distribución geográfica de los usuarios.

♣ Se basan en un mecanismo de composición de EVs de funcionamiento similar a la herencia del paradigma orientado a objetos. Se piensa en un entorno grande en el que puedan existir subentornos con distintas características, las cuales se definen de manera separada y los subentornos *heredan* las características que les interesen. Por otro lado, se ha trabajado con una arquitectura cliente/servidor, en la cual el servidor es en realidad un cluster basado en Linux (Beowulf cluster) o IRIX, lo que además permite hacer uso de multiproceso para atender las acciones que tienen lugar en los clientes. Es de destacar que esta iniciativa se ha incluido dentro de la licencia pública de GNU.

- No se definen los periféricos que se utilizan, aunque se menciona la intención de alcanzar un alto grado de inmersión con el uso de cascos de realidad virtual.
- Posibilidades de Interacción dentro del Sistema:
  - En esta herramienta se busca dotar de un comportamiento complejo y de una capacidad de interacción bastante completa a los usuarios de los entornos que se creen con ella, tanto entre los avatares que conforman el entorno como con los objetos que conformen el EV.
  - Puesto que en realidad estamos hablando de una herramienta de creación de EVs, no existen agentes, pero sí se proporciona el mecanismo necesario para crear agentes con comportamientos complejos con relativa facilidad.
  - Se piensa en un entorno multiusuario con gran número de usuarios.
- No se menciona el uso de ninguna metodología, aunque si se hace referencia a la utilización de lenguajes orientados a objetos para realizar la codificación.

Proyecto MAVERIK [Maverik00]

- Este entorno está pensado para utilizarse en la construcción de EVs.
- Los usuarios típicos serán aquellas personas interesadas en la construcción de EVs, especialmente si lo que buscan es poder utilizar interacciones complejas.
- Se intenta estudiar y probar un método para representar objetos a los que se les puede dar una definición geométrica y con los que, por tanto, se puede interactuar en el EV, como contraposición a los objetos que sólo se representan con texturas.
- Se compone de un *microkernel* y un entorno de desarrollo para facilitar la construcción del EV y de una serie de módulos de soporte que proporcionan funciones para dibujar los objetos del EV, implementar la interacción entre los mismos y dar soporte a distintos tipos de dispositivos de entrada/salida. Es de destacar que está sujeto a la licencia pública GNU.
- Se da soporte a un amplio abanico de dispositivos de conexión: cascos de realidad virtual, reconocimiento de voz, etc.

- Posibilidades de Interacción dentro del Sistema:
  - En principio, está pensado para sistemas monousuario, por lo que no hay interacción con otros avatares, pero, puesto que uno de los objetivos es estudiar la forma de interactuar con los objetos del entorno, el grado de interacción con todos los objetos que aparecen en el EV puede llegar a ser muy alto.
  - La detección de colisiones se realiza con un alto grado de exactitud.
  - Está pensado para construir entornos monousuario; su posterior integración con DEVA hace posible que su utilidad se extienda a entornos multiusuario.
- No mencionan el uso de ninguna metodología de desarrollo.

Proyecto ESCAPE [Escape99]

- Aplicado a la construcción de entornos virtuales a gran escala.
- Está pensado para usuarios que se dediquen a la construcción de EVs.
- Está orientado hacia la investigación de técnicas de construcción de escenarios de grandes dimensiones. Han implementado un EV, Cityscape, con el cual intentan que cada edificio de la ciudad pueda a su vez albergar un EV diferente.
- Se apoyan en Maverik, aunque lo han adaptado para conseguir una mayor velocidad de renderización. Actualmente se está integrando con DEVA para conseguir que el entorno que se construya sea multiusuario.
- Posibilidades de Interacción dentro del Sistema:
  - Las posibilidades de interacción son las definidas para MAVERIK. El objetivo básico es agilizar los métodos proporcionados por éste, ya que son de propósito general. En un futuro, también albergará las capacidades ofrecidas por DEVA.

Tras el estudio de estos proyectos basados en la construcción de EVs, se puede extraer en conclusión que todos ellos tienen como objetivo la construcción de

aplicaciones informáticas, pero sólo enfocadas a la experimentación con diferentes tecnologías *hardware* y con técnicas de renderización y descripción de objetos.

El propósito de este trabajo no es ninguno de los anteriores, sino la aplicación de la Ingeniería de Software como disciplina para la definición y desarrollo de EVs.

2. OBJETIVOS

En el presente trabajo, los EVs objeto de estudio serán aquellos en los que se busca liberar al usuario del mayor número de controles posible, de manera que se tenga una completa libertad de movimientos para desplazarse por el mundo virtual sin preocuparse de realizar muchas acciones ni de manejar muchos controles. Para ello, será necesario dotar a los avatares de comportamiento, basado en una personalidad y un estado de ánimo, según el cual se tomará la decisión de realizar determinadas acciones de una manera concreta.

Como vehículo para estudiar estos EVs, se llevará a cabo el desarrollo de un mundo virtual en el que sea posible jugar al *Escondite Inglés*. Esto permitirá, de forma paralela, estudiar una manera de facilitar a los usuarios el manejo de un avatar por un EV y probar tanto la interacción avatar-avatar como la estructura corporal que necesitan tener los avatares para poder llevar a cabo esta interacción.

Otra de las tareas que se intentarán realizar con el presente trabajo será probar una de las metodologías de desarrollo más en boga actualmente con el fin de determinar si es válida para el desarrollo de este tipo de aplicaciones.

Uno de los problemas que tiene el desarrollo de Entornos Virtuales es que su aparición se produjo en una fecha relativamente reciente, por lo que las metodologías de desarrollo no contemplan aspectos específicos de este tipo de aplicación. En este caso, se intentará comprobar si con el uso de UML (*Unified Modeling Language*), a través del método propuesto por *Larman* [Larman99], es posible realizar de forma completa el desarrollo de un EV y, en caso contrario, determinar qué aspectos del desarrollo no están contemplados en esta metodología.

Se ha pensado en UML básicamente por dos razones:

- El caso que nos ocupa se ajusta muy bien a las metodologías orientadas a objetos, ya que estas metodologías permiten modelar el sistema como un conjunto de objetos que interactúan entre sí. Además, su objetivo es alcanzar la calidad en la construcción del sistema haciendo especial énfasis en la reutilización y la extensibilidad del producto, características fundamentales en la generación de entornos virtuales, ya que la idea es construirlos de forma que

se puedan ampliar fácilmente y se puedan intercambiar componentes entre distintos EVs.

- Como se ha mencionado anteriormente, UML es una metodología que se está implantando de manera muy rápida en todas las organizaciones que realizan desarrollos orientados a objetos, por lo que siempre es una buena idea probarla en distintos dominios para comprobar su utilidad e intentar de alguna manera identificar y completar las posibles carencias de las que pueda adolecer.

Así pues, al finalizar el presente trabajo, se espera, por un lado, haber dejado más clara cuál es la situación actual de los EVs habitados, y por otro, haber comprobado hasta qué punto el método de Larman es adecuado para el desarrollo de este tipo de aplicación. Puesto que Larman ofrece cierta flexibilidad en el orden en que se realizan ciertas tareas, e incluso contempla la posibilidad de no realizar algunas de ellas, este método se va a aplicar siguiendo el esquema que se presenta a continuación:

- Planificación y Especificación de Requisitos.
  - Definición de los Requisitos.
  - Definición de los Casos de Uso en Formato de Alto Nivel.
- Análisis.
  - Definición de los Casos de Uso en Formato Expandido.
  - Definición de los Diagramas de Secuencia del Sistema.
  - Diagramas de Estados.
  - Modelo Conceptual.
  - Definición de los Contratos de Operación.
- Diseño.
  - Definición de la Interfaz de Usuario.
  - Diagramas de Interacción.
  - Diagrama de Clases de Diseño.
  - Esquema de Base de Datos.
  - Refinamiento de la arquitectura del sistema.

- Implementación.

Además, a lo largo del desarrollo, se irán añadiendo nuevos puntos en los apartados en los que se considere que no se contempla algún aspecto de la construcción de EVs. De igual manera, si se considera necesario, se dividirá el desarrollo del EV en varios ciclos.

PLANIFICACIÓN Y ESPECIFICACIÓN DE REQUISITOS

1. ESPECIFICACIÓN DE REQUISITOS

Como primer paso en la construcción de un sistema software, es necesario disponer, de una manera más o menos formal, de una especificación de los requisitos que debe cumplir el sistema, de forma que el desarrollador pueda saber qué es lo que tiene que hacer a la vez que quedan satisfechas las necesidades de quien demandó la construcción del sistema.

Si nos fijamos en el método definido por Larman [Larman99], se parte de la idea de que ya se dispone de un documento de especificación de requisitos, por lo que en ningún momento se describe qué es lo que debe contener este documento o cómo se debe realizar su construcción.

Sin embargo, está claro que, como en cualquier otro sistema, de este o de otro tipo, es necesario disponer, si no de todos, sí de gran parte de los requisitos, pues si no es una tarea harto difícil la construcción de un sistema que no tenga posteriormente que sufrir cambios drásticos o incluso ser descartado para iniciar la construcción de uno nuevo, a la manera de un prototipo.

En cualquier caso, se menciona la posibilidad de no disponer de parte de los requisitos al iniciar la construcción del sistema, sin que ello suponga una merma en la posibilidad de realizar un análisis y un diseño que no sufra grandes modificaciones. Esto es así debido a la fuerte componente gráfica de la que constan estos sistemas. La construcción de los elementos gráficos que conformarán el EV puede realizarse de manera paralela a cualquiera de las fases de desarrollo de la aplicación, lo que posibilita que la especificación de los requisitos que debe cumplir esta parte del sistema se pueda diferir hasta que se haya determinado de una manera más clara qué es lo que se requiere del sistema, si es necesario. Ello no implica, no obstante, que se deba retrasar mucho esta fase, puesto que, con el estado actual de estos sistemas, la implementación de la parte visual de los EV requiere poder ver al instante los resultados que se van alcanzando, y necesita disponer de los modelos que se van a utilizar de manera definitiva, ya que, en general, poseerán características que no podrán ser simuladas por objetos provisionales.

Por estas razones, se ha realizado la especificación de requisitos de la misma forma en que se venía realizando con la metodología estructurada, siguiendo el estándar IEEE 830, según se presenta a continuación.

1.1. PROPÓSITO

El propósito de este desarrollo es el de experimentar con técnicas de interacción sencillas, en tiempo real, que al tiempo que permitan a los usuarios moverse por un entorno virtual, den la posibilidad de popularizar dichos entornos.

Además, se pretende hacer un primer motor de simulación social, es decir, de comportamientos propios de los seres humanos en sus relaciones con otros, a través de su representación virtual. La finalidad es que, si en el futuro los entornos virtuales son puros reflejos de las actividades diarias de los humanos, no perdamos por ello las capacidades de relacionarnos con los demás, aun estando distantes en el espacio.

Por otro lado, la interfaz que se diseñe debe tener en cuenta el hecho de que el tener demasiadas cosas que controlar puede hacer pesado el manejo del avatar. Es por esto por lo que ciertos aspectos del comportamiento de un avatar *se pueden automatizar,* sin que esto repercuta negativamente en el nexo de unión que existe entre un usuario y su avatar, y en cambio sí le facilite enormemente el control de éste en el EV. En resumen, el usuario debe ser capaz de delegar en su avatar aquellas tareas que considere oportunas.

Este grado de automatización llevado a su punto extremo permitirá que el entorno virtual que se desarrolle no sólo albergue avatares manejados por usuarios, sino además agentes autónomos.

Así pues, se tratará de ser lo más sencillos posible, dentro de los márgenes que permite el hecho de construir un sistema de realidad virtual. Por tanto, los objetivos a alcanzar son dos fundamentalmente:

1. Por un lado, implementar una **interfaz sencilla, novedosa y rápida** (puesto que se trata de un sistema de tiempo real).
2. Por otro lado, **analizar, diseñar e implementar el entorno virtual** que dé soporte a nuestros objetivos.

Se analizará y diseñará un entorno virtual multiusuario genérico con las características mínimas necesarias para contener presencia humana virtual, y por último se implementará una primera versión del entorno, donde al menos se introducirá la posibilidad de moverse e interactuar.

Como entorno virtual para la experimentación, se ha pensado en plasmar todos estos propósitos en la realización de un juego virtual en la red Internet. El juego seleccionado es el ampliamente conocido *"Escondite Inglés"*, donde los participantes se ven obligados a moverse por un entorno virtual con un objetivo muy concreto, y a la vez se ven influidos en sus movimientos por el resto de avatares.

Dentro del entorno virtual, habrá que distinguir qué cosas ocurren antes del comienzo del juego y durante el transcurso de éste.

Con el fin de facilitarle al usuario la exteriorización de su humor a través de la expresión de su avatar, se automatizará la forma en que éste exteriorizará su estado de ánimo, en función de las situaciones que vive dentro del entorno virtual.

Por lo tanto, en la conceptualización aparecerán elementos generales, que son aplicables a cualquier entorno virtual de características similares, y otros que son específicos del juego que se va a llevar a cabo dentro del EV. A continuación aparece brevemente descrito el juego del *"Escondite Inglés"*.

1.1.1. Reglas del Escondite Inglés

- El primero que se conecta al juego es el que se situará en una zona concreta, que suele ser una pared, sobre la que se apoya para contar.
- A medida que se vayan conectando al EV, los jugadores tomarán el rol de participantes, que tienen como objetivo el que a continuación se detallará.
- Una vez que se ponen de acuerdo para que comience el juego, cosa que seguramente hagan por medio de Chat, el que se la liga se coloca en la pared en la que contará, y el resto se sitúan en una línea que habrá en el suelo marcando el origen, del que partirán.
- Cada vez que el que se la liga se da la vuelta para contar, sonará una frase como *"Un, dos, tres, al escondite Ingles, sin mover las manos, ni los pies"*. Esto indica que el que se la liga está de espaldas, pero cuando acabe de contar, se dará la vuelta para intentar pillar a todos los participantes que se estén moviendo.
- Mientras el que se la liga cuenta, el resto de participantes pueden moverse hacia el que se la liga, con el objetivo de alcanzar la pared en la que éste está

situado. En el momento en que el que se la liga se vuelve hacia los participantes, todos deberán quedarse inmóviles. Si se mueven y el que se la liga les ve, deberán volver a la línea de origen para empezar de nuevo el juego.

- El primer participante que alcance la pared en la que se encuentra el que se la liga, gana, y le toca ligársela a continuación.

1.2. ALCANCE

El desarrollo que se pretende abordar debe tener una estructura tal que sea lo suficientemente abierto como para la incorporación de diferentes modelos, bien sea de personalidad, sociales, etc.

Además, debe estar preparado para perfeccionar el módulo de captura de movimiento, que es uno de los elementos más susceptibles de cambio.

Este sistema está pensado para ser usado como banco de pruebas para la investigación en el uso de nuevas tecnologías aplicadas a la comunicación, y además, por su sencillez, estará al alcance de mucha gente.

1.3. DEFINICIONES, ACRÓNIMOS Y ABREVIATURAS

A continuación se exponen una relación de términos que se utilizan en esta parte de la documentación.

Término	Definición
Escondite Inglés	Juego que tendrá lugar en el entorno virtual.
Rol	Papel que toman los distintos participantes en el juego.
Jugador	Cualquier usuario que desee jugar.
"Se la liga"	Se dice que se la liga el que está en la pared contando.

A continuación aparecen los acrónimos utilizados en el documento.

Acrónimos y abreviaturas	Definición
MV	Mundo virtual
EV	Entorno Virtual
SS	Simulación Social
3D	Tridimensional
RV	Realidad virtual

Los términos “Mundo Virtual” y “Entorno Virtual” se utilizarán indistintamente en el presente desarrollo.

1.4. CARACTERÍSTICAS DE LOS USUARIOS

Los usuarios que se conecten al EV no han de cumplir ningún tipo de requisito específico. Serán de cualquier edad y sexo; además los conocimientos de informática no tienen por qué ir más allá del simple manejo de un ordenador al nivel de usuario.

1.5. REQUISITOS FUNCIONALES

Debido a las características del sistema a construir, los requisitos se van a dividir según hagan referencia al juego del escondite inglés, a las acciones relativas al humor y la personalidad de los avatares o a las características de los entornos virtuales.

1.5.1. Requisitos del Escondite Inglés

Req(01)	Cuando un jugador se conecte al entorno, existirá una forma de decidir si se la liga o no.
Req(02)	El avatar que se la liga deberá volverse hacia la pared para contar.
Req(03)	Los jugadores detectarán si el avatar que se la liga está contando para poder desplazarse hacia la pared.
Req(04)	Cuando termine de contar, el avatar que se la liga se volverá hacia los jugadores.
Req(05)	Un avatar podrá determinar a qué distancia se encuentra de la pared, decidiendo si está muy lejos o muy cerca.

Req(06)	Un avatar podrá ver si hay otros avatares que se encuentren por delante de él.
Req(07)	El avatar que se la liga, cuando mire a los jugadores, podrá detectar si se están moviendo.
Req(08)	Si un avatar ha sido visto moviéndose, se le deberá comunicar de manera expresa.
Req(09)	Un avatar podrá decidir si le han visto moviéndose demasiadas veces.
Req(10)	Un avatar será capaz de decidir si la duración de la partida es demasiado larga.
Req(11)	Un avatar detectará que ha ganado la partida cuando llegue el primero a la pared.
Req(12)	Los avatares serán capaces de ver que otro avatar ha ganado la partida.
Req(13)	Cuando un avatar vea que otro ha ganado la partida, sabrá que él la ha perdido.
Req(14)	Cuando se acabe una partida, los jugadores deberán volver a la línea de comienzo.
Req(15)	Los avatares podrán correr de frente.
Req(16)	Los avatares podrán andar de frente y de espaldas.

1.5.2. Requisitos del Modelo de Humor y Personalidad

Req(17)	El usuario podrá elegir los rasgos de la personalidad que tendrá su avatar dentro del EV.
Req(18)	Los avatares tendrán unos parámetros que definan su estado de ánimo.
Req(19)	Se definirá un comportamiento para los avatares según sus rasgos de personalidad y su estado de ánimo.

Req(20)	Los avatares podrán saludar a otros avatares.
Req(21)	Un avatar podrá detectar si otro avatar le está saludando.
Req(22)	Un avatar podrá hacer reír a otros avatares.
Req(23)	Un avatar podrá darse cuenta de que otro avatar está intentando hacerle reír.
Req(24)	Los avatares podrán ponerse tristes.
Req(25)	Un avatar será capaz de ver si otros avatares están tristes.
Req(26)	Un avatar podrá enfadarse.
Req(27)	El estado de enfado de un avatar podrá ser percibido por otros avatares.
Req(28)	Los avatares podrán sentirse aburridos.
Req(29)	Un avatar podrá detectar si otro avatar está aburrido.
Req(30)	Un avatar será capaz de darse cuenta de si tiene otros avatares alrededor.
Req(31)	Un avatar podrá agredir a otro.
Req(32)	Un avatar que ha sido agredido deberá mostrar físicamente el resultado de la agresión.
Req(33)	Los avatares podrán reírse.
Req(34)	Los avatares podrán saltar de alegría.
Req(35)	Un avatar podrá bailar.
Req(36)	Un avatar podrá patalear.
Req(37)	Un avatar podrá llorar.

1.5.3. Requisitos de los EVs

Req(38)	El usuario deberá poder elegir a qué servidor de simulación se conecta al entrar en el EV.
Req(39)	El usuario podrá seleccionar el avatar con que quiere ser representado en el EV de entre todos los disponibles.
Req(40)	El usuario podrá elegir el punto de vista que desea utilizar para moverse por el MV.
Req(41)	El usuario podrá elegir el dispositivo de detección de movimiento que desea utilizar, siendo una de las posibilidades el teclado.
Req(42)	El avatar se podrá mover con total libertad por todo el escenario que conforme el EV.
Req(43)	Los avatares podrán girarse un determinado número de grados.
Req(44)	Los avatares podrán reconocer a otros avatares con los que hayan coincidido en otras conexiones al EV.
Req(45)	Los avatares podrán detectar puertas por las cuales pueden abandonar el EV.
Req(46)	Los avatares podrán detectar obstáculos que les impidan desplazarse por el EV.
Req(47)	Los usuarios, a través de un chat, podrán comunicarse con otros usuarios.
Req(48)	El EV se deberá renderizar cada cierto tiempo para que los usuarios puedan ver los cambios que se van produciendo.

1.6. REQUISITOS DE INTERFAZ

1.6.1. Interfaz con el usuario

Las salidas del sistema van destinadas al usuario del mismo, y serán ofrecidas a través de la pantalla, con la posibilidad de ampliar ésta y proyectar el contenido del EV sobre una pared.

Además, el usuario podrá elegir entre dos tipos de interfaz para comunicarse con el ordenador, bien con teclas y botones, como se hace habitualmente, o bien con un dispositivo de captura de movimiento.

Como interfaz entre el EV y el usuario se podrá seleccionar entre dos posibilidades:

- Un teclado corriente, con el cual el usuario podrá ordenar a su avatar que ande, corra o se pare.
- Un dispositivo de captura de movimiento, que en este caso será una cámara de vigilancia continua con ojo de pez (mínima distancia, máximo ángulo de visión).

1.6.2. Interfaz con otros sistemas

Vamos a hacer especial hincapié en que existirá un sistema de captura de movimiento, que por supuesto será opcional, y que muy posiblemente se trate de un dispositivo de captura continua.

1.6.3. Selección del dispositivo de captura de movimiento

Otro punto importante a considerar aquí es el problema de seleccionar un dispositivo de captura de movimiento. Es preciso realizar la selección cuanto antes, ya que el desarrollo e implementación de otros muchos elementos puede venir determinado por el sistema de captura que elijamos.

1.6.3.1. Interfaces de Realidad Virtual basado en captura de movimiento

Hoy en día existen básicamente dos técnicas para capturar la posición de un cuerpo completo en tiempo real. Una utiliza ***cámaras de vídeo,*** las cuales devuelven imágenes que pueden ser tratadas con el fin de localizar la posición de los miembros del cuerpo. Esta técnica ha sido utilizada con éxito en el sistema ALIVE (Maes, Darrel, et al, 1995), donde se usa la imagen del usuario para extraer información de varias partes del cuerpo.

La segunda técnica se basa en la utilización de ***sensores***, los cuales van pegados al usuario. Los más comunes son los que miden un campo magnético generado en un punto de referencia. El valor de la medición se transforma en las correspondientes

coordenadas de posición y orientación, y son enviadas al ordenador. Estos datos se emparejan con los puntos de rotación de un esqueleto virtual, a través de un convertidor anatómico.

1.6.3.2. Interfaz seleccionada para el EV

De las dos técnicas de captura de movimiento mencionadas, se ha decidido utilizar la primera de ellas, por cuestiones de sencillez, de economía, y de comodidad para el usuario, a pesar de que sufre de restricciones de visibilidad, relacionadas con la cámara, y de que el módulo de visión para la extracción de información proveniente de la cámara es muy delicado, puesto que ha de funcionar con tiempos de respuesta muy pequeños, y requiere desarrollar un software especial para el análisis de la imagen capturada. Además, con el fin de no perder detalle de la escena, la cámara ha de ser de captura continua, aunque no se analicen todos los fotogramas que captura.

Por otro lado, la cámara que se elija ha de tener un angular muy grande, con el fin de que el usuario no necesite un espacio enorme para reproducir el escenario necesario, sino que pueda reconstruirlo en una habitación o en un pasillo de su casa.

La ventaja que tiene el usar este tipo de dispositivos es que como el usuario no lleva ningún cable, le resulta mucho más cómodo moverse, y éste es uno de los objetivos marcados desde un principio: liberar al usuario del mayor número de controles y de cables.

El hecho de seleccionar este tipo de sistemas de captura de movimiento lleva implícito que el sistema de RV sea del tipo *WoW (Window on World)*, también conocidos como *Desktop VR*. Este tipo de sistemas se clasifican dentro de los de "No-inmersión", debido a la naturaleza de los dispositivos utilizados. En este tipo de sistemas, el usuario tiene una ventana a través de la cual ve el EV. Esta ventana puede ser simplemente la pantalla del ordenador.

En el caso concreto que nos ocupa, el hecho de que el usuario utilice un sistema de captura de movimiento implica que tiene libertad para moverse por un espacio cuyas dimensiones estarán limitadas por lo que el angular de la cámara sea capaz de recoger. Puesto que la pantalla del ordenador es un espacio bastante reducido para ver el EV, proponemos utilizar, como ventana al EV, una pantalla grande, con el fin de que el EV sea del mismo tamaño que el mundo real, de manera que el usuario no tenga que hacerse pequeño para estar en el EV, sino que éste sea grande para acercarse al usuario.

El escenario propuesto es el que aparece en la Figura 1.

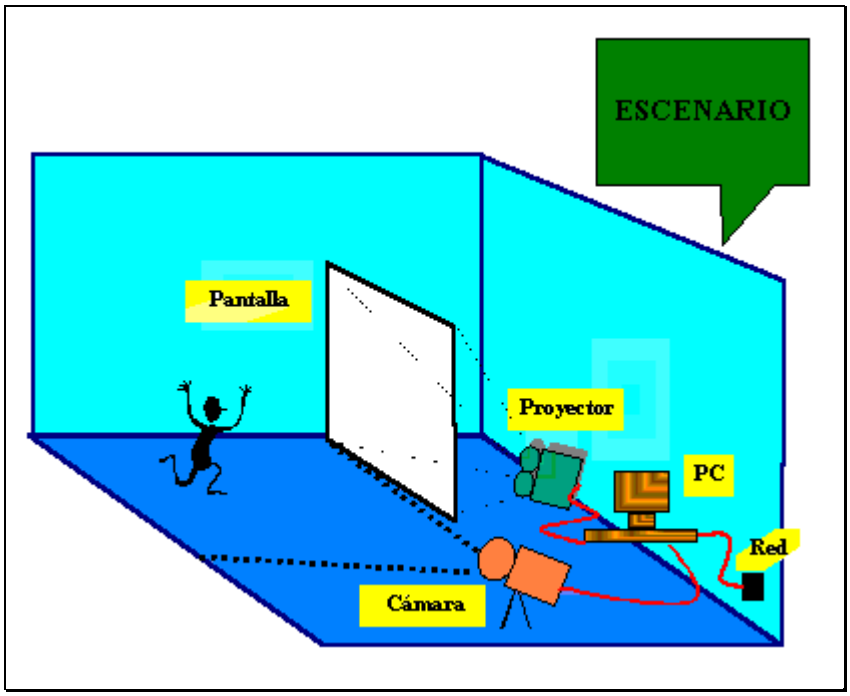

Figura 1: Escenario en la vida real

A continuación se describen los elementos necesarios para construir el escenario real:

- Una pantalla grande y un proyector. Ambos se pueden sustituir, por ejemplo, por un televisor muy grande. En cualquier caso, la opción seleccionada representará la ventana al EV. A pesar de que nuestro sistema se clasifica dentro de los de No-inmersión, creemos que el grado de inmersión de un sistema de realidad virtual se puede mejorar utilizando la técnica apropiada; ésta es la razón de utilizar una pantalla grande, el intentar que el usuario se sienta más cerca del EV.

- Una ***cámara***: en este trabajo nos proponemos simplemente capturar el desplazamiento del usuario en el espacio real, para después representarlo en el EV. Necesitaremos una cámara de vigilancia, ya que la captura de movimiento ha de ser continua.

- Un ***ordenador personal***: en el que tendremos el cliente del EV ejecutándose. Éste tendrá una interfaz con la cámara, a través de la cual sabrá si ha habido desplazamiento del usuario o no. Además ha de tener conexión a la red.

1.7. REQUISITOS NO FUNCIONALES

1.7.1. Requisitos Hardware

Con el fin de que este sistema sea lo más accesible posible, se propone que el equipo del usuario final sea un ordenador personal. Como la aplicación va a ser accesible a través de red, será imprescindible disponer del hardware necesario para acceder a una red TCP/IP del tipo de INTERNET.

En el presente desarrollo hay una serie de puntos que marcan en gran medida las decisiones que se tomen, en cuanto a hardware se refiere:

1. Puesto que se trata de un sistema multiusuario, tendremos un servidor que gestione las conexiones. Este servidor debe ser lo suficientemente potente para tal fin.
2. El diseño gráfico de los modelos 3D que componen el entorno virtual, así como de los avatares.
3. La renderización del entorno virtual, así como de todos los elementos que lo componen y de los avatares.
4. La captura, en tiempo real, del movimiento del usuario, si es que ha decidido utilizar una cámara de captura de movimiento.

Todos los elementos antes mencionados, junto con las elecciones de software, determinarán las características deseables para el ordenador personal del usuario final.

En el caso de usar el dispositivo de captura de movimiento mencionado en el apartado anterior, necesitaremos tener instalada en el PC una tarjeta digitalizadora de vídeo, con el fin de poder analizar las imágenes que toma la cámara de vigilancia.

Además será necesaria una tarjeta de sonido para escuchar al avatar que se la liga mientras habla.

A continuación se describen los dispositivos que han de tener unas características concretas; el resto de ellos pueden ser de cualquier marca y tipo.

1.7.1.1. Descripción de la cámara

Auto-Iris Fish-Eye Vari-focal lens A/I 1'6-3'4 mm
F14 ½ Pulgadas
Fabricante: Cosmicar/Pentax
Eutaric: 9002190090

Con esta cámara tenemos hasta un ángulo de 180 grados. Este tipo de cámaras resulta idóneo para lugares en los que la distancia entre la cámara y el objeto a observar no puede ser excesivamente grande.

1.7.1.2. Tarjeta DT3153

La cámara recogerá una imagen que representa lo que está ocurriendo en el mundo real, de modo que será necesaria una tarjeta digitalizadora. Concretamente utilizaremos una tarjeta "Data Translation's DT3153 Frame Grabber". Se trata de una tarjeta basada en un bus PCI de alta precisión, con controladores para Windows 95/98 y NT. El DT3153 es ideal para análisis de imágenes en color.

1.7.1.3. Máquinas de desarrollo

Como máquinas de desarrollo de utilizarán:

- Silicon Graphics O2, para el desarrollo de los modelos tridimensionales y las pruebas iniciales de animación.
- PCs para el posterior ensamblaje de los modelos y la programación del comportamiento del EV.

1.7.2. Requisitos Software

La aplicación deberá funcionar en ordenadores con sistema operativo Windows NT y Windows 98, pero se desarrollará sobre Windows NT, por las razones que se especifican en el apartado siguiente de alternativas de construcción.

1.7.2.1. Software de diseño gráfico

Será necesario diseñar los modelos 3D para los avatares así como el resto de elementos que conformen el EV. Los productos de uso más frecuente que existen actualmente en el mercado son: Alias WaveFront PowerAnimator, 3DStudioMax y

SoftImage. *Alias WaveFront* y *SoftImage* están disponibles en múltiples plataformas, y *3DStudio Max* está disponible en PC.

1.7.2.2. Herramientas de desarrollo

Para realizar el diseño y posterior implementación de un EV hay que tener muy en cuenta sus características con el fin de realizar una correcta elección de la herramienta que se va a utilizar para realizar la implementación. A lo largo de este apartado se van a presentar las características que se han tenido en cuenta a la hora de realizar la elección de una herramienta de trabajo. Finalmente, se presentará una descripción más detallada de la herramienta seleccionada para plasmar el desarrollo que se va a realizar a lo largo del presente documento.

A la hora de implementar un EV nos vamos a encontrar con que disponemos de dos grandes grupos de herramientas para alcanzar nuestro cometido. Por un lado, contamos con librerías gráficas de programación, las cuales incorporan una serie de funciones que facilitan el manejo de objetos 3D con más o menos facilidad, incorporan el manejo de luces y, en definitiva, proporcionan un control suficiente como para poder realizar lo que se desee, eso sí, empleando gran cantidad de tiempo y grandes dosis de trabajo. Por otro lado, y en menor cantidad, existen entornos integrados de desarrollo que se apoyan en las citadas librerías y que, si bien facilitan enormemente el proceso de desarrollo, también permiten un control más pobre sobre las acciones que se quieren realizar.

Las herramientas que se han estudiado para realizar la implementación han sido:

- DirectX: librería de programación desarrollada por Microsoft que incluye manejo de gráficos 2D y 3D, música y sonido, dispositivos de entrada/salida y conexión a la red. En el apartado de gráficos 3D (Direct3D), incluye un módulo de manejo a muy bajo nivel, el Immediate Mode, que se fija en el funcionamiento de la tarjeta gráfica y vértices y normales de las mallas que componen un objeto, y otro que se apoya sobre este, el Retained Mode, que, sin ser de alto nivel, permite abstraer un poco la representación de los objetos y tratar directamente con jerarquías entre los mismos, texturas que se les aplican o rotaciones y traslaciones sin necesidad de recurrir a productos matriciales. Está disponible sólo para el sistema operativo Windows, y sólo se

puede utilizar con el lenguaje de programación C++ y, en su última versión (la 7.0) con Visual Basic.

- ♣ OpenGL: librería desarrollada por Silicon Graphics, se encuentra disponible para un amplio número de plataformas: IRIX, Windows, Linux... Se trata de una librería de programación de bajo nivel, comparable al Immediate Mode de DirectX.
- ♣ Open Inventor: librería de programación que se apoya sobre OpenGL y que constituye el equivalente al Retained Mode de DirectX.
- ♣ Glide: librería de programación desarrollada por 3DFX Interactive, se encuentra al mismo nivel que OpenGL o Direct3D Immediate Mode, aunque su uso se encuentra mucho menos extendido que el de éstas.
- ♣ WorldToolkit: librería de programación desarrollada por Sense8, se encuentra algo por encima de Direct3D Retained Mode y Open Inventor. Está más orientada a la construcción de EVs que las anteriores, las cuales se ocupan del manejo de gráficos 3D en general. Su principal inconveniente se centra en el elevado precio que hay que pagar por las licencias de utilización y distribución.
- ♣ WorldUp/W2W: entorno gráfico de desarrollo creado también por Sense8, se apoya sobre WorldToolkit para proporcionar una herramienta de desarrollo rápido que, si bien es bastante fácil de manejar, ofrece muy poco control sobre los objetos 3D y produce un EV de ejecución sensiblemente más lenta que los anteriores. Cuenta con la gran ventaja de que es enormemente sencillo crear EVs multiusuario sin necesidad de preocuparse en absoluto por la red.

A continuación se van a exponer tres de la características más importantes en las que nos hemos fijado para elegir entre las herramientas descritas anteriormente.

1.7.2.2.1. Lenguajes interpretados vs. lenguajes compilados

Esta característica va a venir dada, en general, por el tipo de herramienta de desarrollo que se utilice para construir el EV. Cuando se utiliza un entorno integrado de desarrollo, lo que se produce es un fichero no ejecutable directamente, por lo que suele ser necesaria una aplicación para interpretar el contenido del archivo que contiene el EV. En este tipo de EV, para dotar de comportamiento a cualquier objeto se va a tener

que recurrir a trozos de código que no van a estar compilados, sino que se van a tener que interpretar cada vez que se ejecuten. Esto presenta la desventaja de que un intérprete siempre será más lento en ejecutar que algo ya compilado y traducido a código máquina, pero presenta la ventaja de que es más rápido a la hora de desarrollarlo y, en general, será más fácil añadir o eliminar ciertas acciones asociadas a un objeto.

En cuanto a las librerías de programación, todo el código que se genere será compilado e incluido en el ejecutable, salvo que se implemente un intérprete similar a los anteriores. Los lenguajes compilados presentarán la ventaja de que su ejecución es mucho más rápida que en el caso de los interpretados, pero, en contrapartida, cada vez que se quiera cambiar el comportamiento de algún objeto haya que tocar el código, será necesario volver a compilarlo y a generar el ejecutable, lo cual no será factible en todas las ocasiones.

1.7.2.2.2. Técnicas de construcción de animaciones

En lo que respecta a la construcción de animaciones, también nos encontraremos ante dos grandes grupos, como en el caso del tipo de lenguaje, pero esta vez no va a depender del tipo de herramienta de desarrollo, sino de cada herramienta en concreto.

La diferencia entre las dos técnicas de animación va a venir dada, en principio, por el hecho de que se disponga de algún mecanismo para animar objetos o haya que programarlo.

En el caso de tener que programarlo, habrá que determinar los puntos por donde hay que mover los objetos y habrá que controlar cuándo se realiza el movimiento.

En las herramientas que proporcionen algún mecanismo para producir animaciones, podremos encontrar dos casos. En el primero de ellos, nos limitaremos a determinar los puntos por donde se mueve el objeto, y la herramienta se encargará de moverlos cuando lo considere conveniente, en general con una velocidad constante. Por este motivo, la velocidad del movimiento dependerá de la distancia a la que situemos los diferentes puntos por los que debe pasar el objeto. En el segundo caso, indicaremos los puntos por los que queremos que pase el objeto y se podrá indicar a la herramienta que, usando algún tipo de interpolación, defina puntos intermedios para que el movimiento no sea tan brusco. De nuevo, nos encontramos ante dos casos posibles. En uno, el número de puntos intermedios será fijo, y no habrá que controlar cuándo se realiza un movimiento, con lo que nos encontramos en el primero de los casos definidos

anteriormente. En el otro, el número de puntos se determinará durante la ejecución, pero habrá que controlar cuándo se producen los movimientos. Ninguna de las aplicaciones que se han analizado presenta los dos últimos métodos que se han descrito.

En resumen, tenemos las siguientes situaciones:

- Sin mecanismos de animación.
- Con mecanismos de animación:
  - Sin interpolación.
  - Con interpolación:
    - Número de puntos intermedios fijo.
    - Número de puntos intermedios variable.

1.7.2.2.3. Técnicas de detección de colisiones

La detección de colisiones será una de las labores en las que haya que poner más cuidado a la hora de implementar el EV, ya que de su correcto funcionamiento dependerá en gran parte el realismo del mismo. La detección de colisiones trata de hacer que cuando haya un objeto que represente un obstáculo para el movimiento de otro, este último no se limite a atravesar al primero, sino que se pare antes de que esto suceda. Sin embargo, también hay que intentar que se pare lo suficientemente cerca como para que no parezca que el sistema no funciona bien y que el objeto se ha parado sin motivo aparente.

Las dos técnicas existentes para realizar la detección de colisiones son:

- *Bounding boxes*: cuando dibujamos un objeto tridimensional, para simplificar ciertas acciones, podemos considerar que se encuentra metido en una caja de las dimensiones mínimas necesarias para albergarlo dentro, la que se conoce como *bounding box*. Sin embargo, la forma en que el objeto está metido dentro de la caja depende de su posición respecto de los ejes de coordenadas. Los lados de la caja siempre son paralelos a los ejes de coordenadas, pero no tiene por qué suceder lo mismo con el objeto, razón por la cual se nos pueden presentar las dos situaciones que se reflejan en la Figura 2.

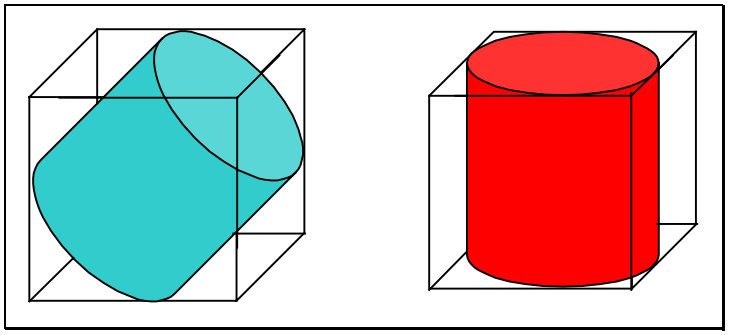

Figura 2: Posición de la *bounding box* respecto a un objeto

La técnica de detección de colisiones basada en *bounding boxes* se centra en la comprobación de si la caja que rodea un objeto choca con la que rodea otro objeto. Si bien el método es bastante rápido, es fácil ver que presenta un grave inconveniente. En el caso de que tuviésemos una pieza que representa una pared con el hueco de una puerta, el hueco sería considerado como parte de la pared, por lo que al intentar atravesarlo se detectaría una colisión y no podría continuar el movimiento. Una posible solución sería formar la pared con tres piezas, con la consiguiente carga que supondría para el sistema, pues para la misma pared estaríamos triplicando el número de puntos que la forman y el número de objetos que hay que tener en cuenta.

- *Ray-tracing*: este mecanismo se basa en, dados un punto y una dirección, trazar un rayo que parta del punto, que siga la dirección dada y vaya comprobando si se choca o no con algún objeto. Si se choca, además, devolverá la distancia a la que se encuentra el objeto, para que decidamos si debemos pararnos o si todavía puede continuar el movimiento. Presenta la ventaja de ser un mecanismo mucho más exacto que el anterior, y a la vez constituye una de sus principales desventajas, ya que, por una parte, los cálculos son más complejos y más lentos, y por otra, podría darse el caso de que, al encontrarnos ante una pared con agujeros de pequeño tamaño, el rayo atraviese uno de esos agujeros y no detecte la presencia de la pared, con lo cual se atravesaría la misma.

1.7.2.3. Descripción de la herramienta de desarrollo seleccionada

De todas las herramientas analizadas, la que se ha elegido para realizar la implementación ha sido WorldUp (Sense8®). El motivo principal ha sido que esta

herramienta proporcionará la posibilidad de realizar una implementación relativamente rápida y fiable.

Como ya se ha dicho anteriormente, WorldUp es un entorno integrado de desarrollo que presenta una serie de características muy interesantes a la hora de construir un EV.

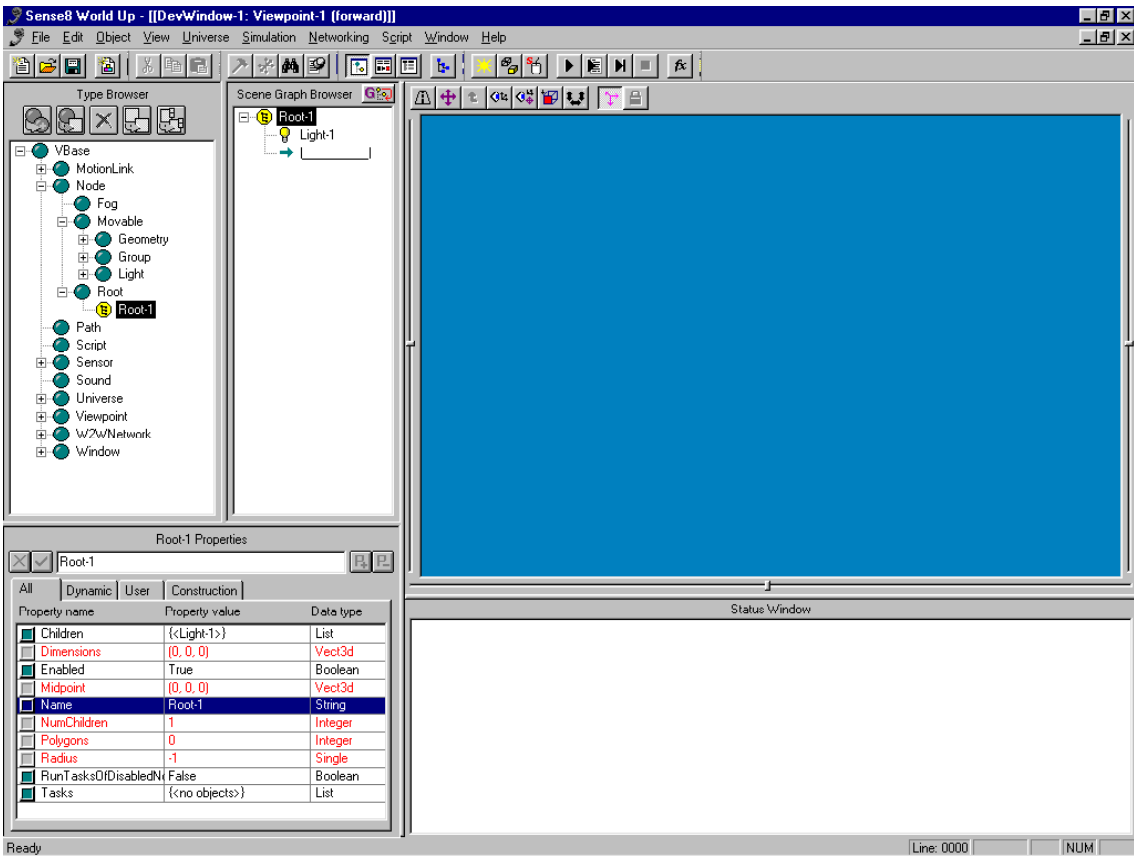

Figura 3: El entorno de desarrollo WorldUp

Por una parte, es posible leer los modelos realizados con Alias Power Animator, tras una conversión del formato de los archivos que genera esta herramienta de modelado 3D, o de 3DStudio, respetando además las jerarquías que se hayan podido establecer entre las piezas de los modelos importados, así como las texturas que pudiesen tener.

Por otro lado, es una característica muy interesante el que ya cuente con una implementación de la conexión en red y que exista un servidor, World2World, que pueda conectar distintos clientes entre sí, sin más que indicar una dirección IP y un puerto de conexión y marcar las distintas propiedades que se quieren compartir por la red.

1.7.2.3.1. Scripts

Otra característica con la que cuenta WorldUp es el hecho de disponer de un lenguaje interpretado, BasicScript, para determinar el comportamiento de cualquier objeto del EV. Estos comportamientos se programan en *scripts* que luego se asocian al objeto en cuestión, de manera que si el objeto no tiene el *script* asociado no presentará ese comportamiento.

Básicamente, existen dos tipos de scripts que se podrán utilizar cuando se implemente el EV:

- Autónomos: se ejecutan bajo demanda, y se encargan de tareas de carácter general, como ejecutar las acciones que provoca la pulsación de un botón del interfaz de la aplicación.
- Tareas: se asocian a los objetos visibles del EV, y se intentan ejecutar de manera periódica. Como es posible asociarlos y desasociarlos de los objetos de forma dinámica, este comportamiento se puede modificar a voluntad.

1.7.2.3.2. Paths

WorldUp ofrece bastantes de las técnicas de creación de animaciones que se han explicado anteriormente (artesanal, sin interpolación y con interpolación y número de puntos intermedios fijo) y las dos técnicas de detección de colisiones que se han mencionado (*bounding boxes* y *ray-tracing*). Para almacenar las animaciones posee un tipo especial de objeto que recibe el nombre de *path* y que almacena la posición y la orientación del objeto que está siendo animado a lo largo de la misma. Si se quieren realizar animaciones usando interpolación, se ofrecen tres posibilidades:

- Lineal: los puntos del *path* se unen mediante líneas rectas.
- B-Splines: el *path* se transforma en una curva bastante suave que, en general, no pasa exactamente por los puntos definidos para éste.
- Bezier: el *path* se transforma en una curva algo menos suave que la anterior pero que sí pasa por los puntos que se han definido para el mismo.

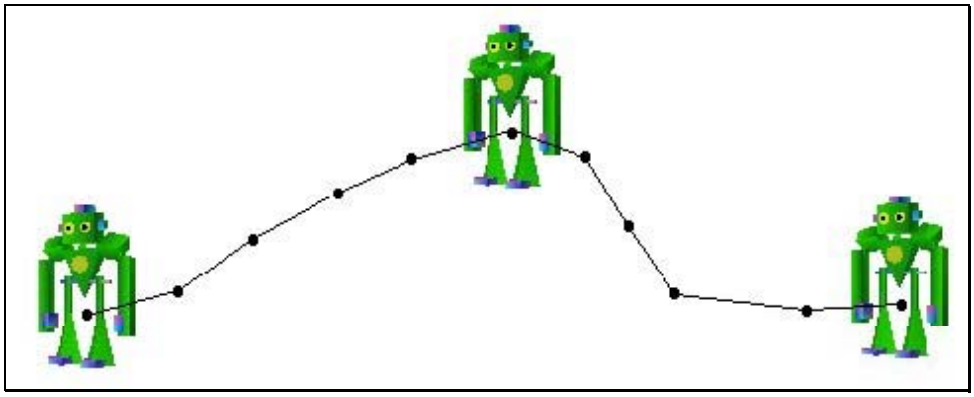

Figura 4: Formato de un *path*

Es posible cambiar la velocidad de ejecución de estos *paths*, así como el sentido en que se reproducen o si se reproducen de manera continuada, lo cual facilita la creación de animaciones de complejidad bastante elevada.

1.7.2.3.3. Otras Características

Otras características que hacen que este entorno resulte especialmente útil es el hecho de que el entorno de desarrollo cuenta con una serie de objetos predefinidos que sirven para iniciar la construcción, así como que el entorno de trabajo está estructurado para trabajar con el manejo de objetos, lo cual permite un aprovechamiento total de un diseño orientado a objetos, pues permite asimilar los objetos del diseño con los objetos del entorno de desarrollo. También es posible hacer uso de librerías dinámicas programadas en C o C++, de manera que se pueda extender el uso de dispositivos o características que no proporciona WorldUp directamente.

Aunque no es posible crear nuevos tipos de objetos o modificar los existentes, sí es posible crear tipos de objetos que se deriven de los existentes, a los cuales se les podrán añadir nuevas propiedades, lo cual nos servirá para crear todos los tipos de objetos que necesitemos para implementar el EV. En la Figura 5 se puede observar cuál es la jerarquía de objetos predefinidos de WorldUp.

A continuación, en la Figura 6, se muestra cuál es el orden que sigue el entorno de desarrollo para ejecutar todas las tareas que se tienen que llevar a cabo dentro del mismo.

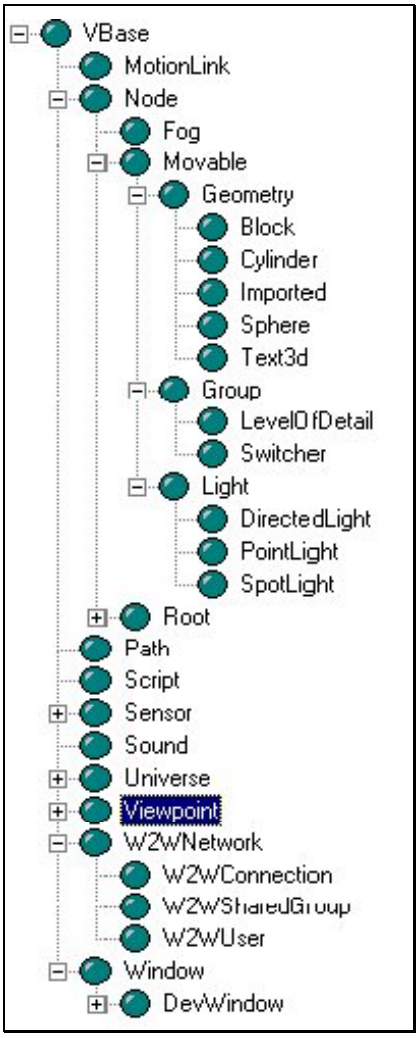

Figura 5: Jerarquía de objetos predefinidos

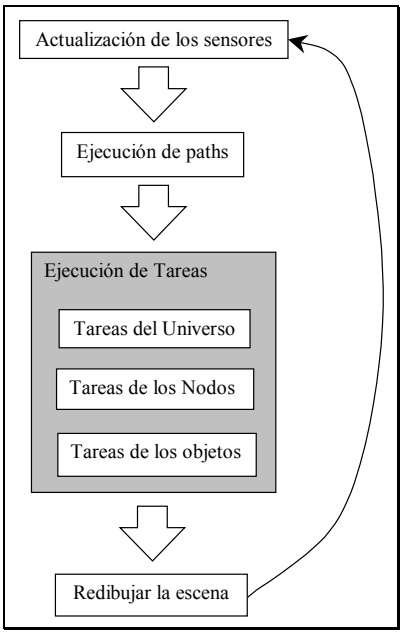

Figura 6: Bucle de Ejecución

1.7.3. Selección de Hardware y Software para el desarrollo

La tarea de diseño gráfico, tanto de los elementos que componen el entorno virtual como de los habitantes de éste, se realizará con Alias Wavefront usando como plataforma Silicon Graphics.

Los modelos 3D serán importados desde Alias Wavefront, y tanto las animaciones como los procedimientos de interacción serán programados usando WorldUp.

El EV resultante funcionará en ordenadores personales. La conexión entre los diferentes usuarios se hará utilizando la plataforma que proporciona W2W, puesto que así se evita tener que desarrollar el software propio de la conexión.

La combinación *WorldUp-World2World* parece que contempla por sí sola una serie de elementos clave en el marco de la creación de EV multiusuario:

- Resuelven el manejo de objetos, de forma sencilla, dentro del entorno de la aplicación.
- WorldUp es interpretado y proporciona un entorno basado en ventanas, lo cual facilita el aprendizaje y minimiza las horas de trabajo de los desarrolladores.
- Ambas se integran con el fin de facilitar las conexiones al EV. Además toda la transmisión corre a cargo de World2World.
- WorldUp incorpora un sistema de Chat, de modo que también tenemos resuelto otro de los requisitos de comunicación entre los usuarios.
- Permiten la conexión de sistemas de captura de movimiento.

2. DEFINICIÓN DE LOS CASOS DE USO

Este es el primer paso que aparece definido dentro del método de Larman [Larman99]. Según definió Jacobson [Jacobson92], un caso de uso es un documento narrativo que describe la secuencia de eventos de un actor (agente externo) que usa un sistema para completar un proceso, es decir, una forma de usar una función que ofrece el sistema.

Cuando nos fijamos en la construcción de un EV, una metodología como esta, que está dirigida por casos de uso, presenta un grave inconveniente: no tiene en cuenta el hecho de que pueden existir agentes autónomos que pueden realizar un gran número de funciones cuyo control no se le ofrece al usuario. Esto da lugar a que estas acciones no aparezcan dentro de los casos de uso, con lo cual no se podrían introducir en el desarrollo del sistema, y no aparecerían en el resultado final funciones que se pedían explícitamente en los requisitos iniciales.

Podría pensarse en introducir en esta etapa unos diagramas de estado que definan el comportamiento de estos agentes autónomos, pero nos encontramos con el problema de que aún no se conoce ninguna característica del sistema, por estar todavía en una etapa del desarrollo muy temprana, razón por la cual resulta imposible definir estados coherentes que permitan modelar el comportamiento de los citados agentes.

También podría darse el caso de que el usuario pueda delegar algunas de las funciones que tiene que llevar a cabo en estos mismos agentes, lo que llevaría a tener que definir por duplicado la realización de una acción determinada, según la controle el usuario o la haya delegado en el sistema, de manera que sea un elemento software el que se encargue de decidir cuándo se realiza esa acción. Este caso presenta menos complicaciones que el anterior, ya que en un primer ciclo de desarrollo podría definirse la manera en que el usuario realiza esta acción, y en un ciclo de desarrollo posterior ya se contaría con datos suficientes como para plantear un diagrama de estados que permita conocer cuál será el comportamiento del agente.

En cualquier caso, y debido a que habría que definir de alguna forma estas funciones del sistema para su posterior inclusión en una planificación de casos de uso en ciclos de desarrollo para la construcción del sistema, se hace necesario definir de alguna manera cuáles serán dichas funciones, para lo cual se propone una notación en forma de conceptos de uso según se describe en [Sánchez99].

2.1. DIAGRAMA DE CASOS DE USO

En este punto y en el siguiente, se van a describir los casos de uso que determinan las acciones que podrá demandar el usuario al sistema, reflejando de igual modo cuál es el grado de control que puede tener un usuario sobre su avatar.

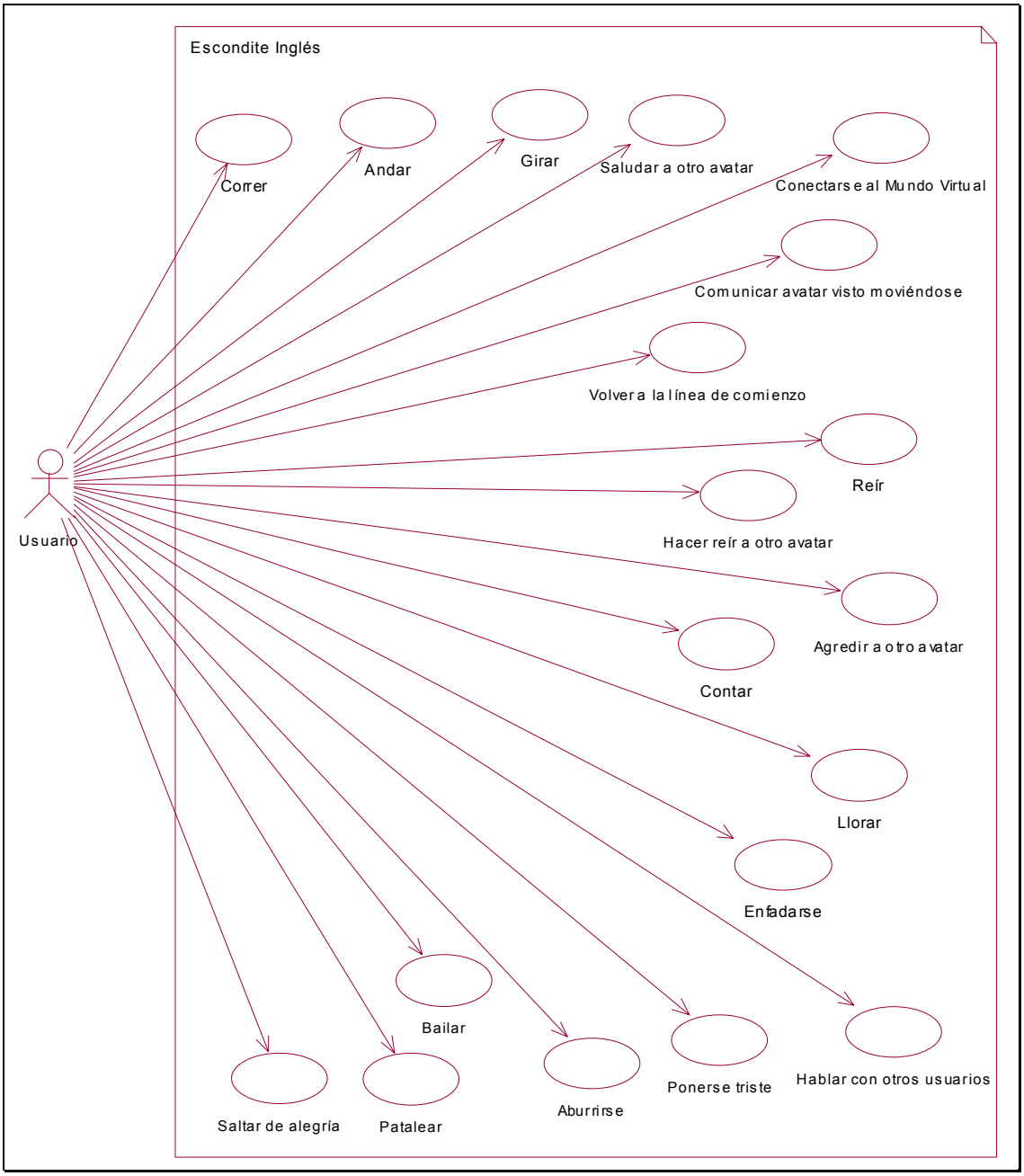

Figura 7: Diagrama de casos de uso

2.2. CASOS DE USO DE ALTO NIVEL

Caso de Uso Conectarse al Mundo Virtual

Actores Usuario

Tipo Primario

Descripción El usuario inicia la ejecución del EV. El sistema le pide que seleccione un avatar, un punto de vista, los rasgos de personalidad de su avatar, un servidor de conexión al EV y un dispositivo de captura de movimiento. El usuario introduce todos los datos pedidos y el sistema realiza la conexión al EV, asignándole un rol al jugador.

Caso de Uso Contar

Actores Usuario

Tipo Primario

Descripción El jugador que se la liga indica al sistema que quiere empezar a contar. El sistema indica al avatar que se dé la vuelta hacia la pared; tras esto, indicará al avatar que cuente y, por último, que se dé la vuelta hacia los jugadores.

Caso de Uso Saludar a otro avatar

Actores Usuario

Tipo Primario

Descripción El usuario indica al sistema que quiere saludar, y éste le indica al avatar que representa a este usuario que realice la acción de saludar que esté de acuerdo con su personalidad y su humor.

Caso de Uso Hacer reír a otro avatar

Actores Usuario

Tipo Primario

Descripción El usuario indica al sistema que quiere hacer reír a otro avatar. El sistema le indica al avatar que representa a este usuario que realice la acción de hacer reír que esté de acuerdo con su personalidad y su humor.

Caso de Uso Agredir a otro avatar

Actores Usuario

Tipo Primario

Descripción El usuario indica al sistema que quiere agredir a otro avatar. El sistema le indica al avatar que representa a este usuario que realice la acción de agredir que esté de acuerdo con su personalidad y su humor.

Caso de Uso Reír

Actores Usuario

Tipo Primario

Descripción El usuario indica al sistema que quiere que su avatar se ría. El sistema ordena al avatar que representa al usuario que realice la acción de reír que esté de acuerdo con su humor y su personalidad.

Caso de Uso Llorar

Actores Usuario

Tipo Primario

Descripción El usuario indica al sistema que quiere que su avatar llore. El sistema ordena al avatar que representa al usuario que realice la acción de llorar que esté de acuerdo con su humor y su personalidad.

Caso de Uso Ponerse triste

Actores Usuario

Tipo Primario

Descripción El usuario indica al sistema que quiere que su avatar se ponga triste. El sistema ordena al avatar que representa al usuario que realice la acción de ponerse triste que esté de acuerdo con su humor y su personalidad.

Caso de Uso Enfadarse

Actores Usuario

Tipo Primario

Descripción El usuario indica al sistema que quiere que su avatar se enfade. El sistema ordena al avatar que representa al usuario que realice la acción de enfadarse que esté de acuerdo con su humor y su personalidad.

Caso de Uso Aburrirse

Actores Usuario

Tipo Primario

Descripción El usuario indica al sistema que quiere que su avatar adopte la posición de aburrirse. El sistema ordena al avatar del usuario que realice la acción de aburrirse que esté de acuerdo con su humor y su personalidad.

Caso de Uso Saltar de alegría

Actores Usuario

Tipo Primario

Descripción El usuario indica al sistema que quiere que su avatar salte de alegría. El sistema ordena al avatar que representa al usuario que realice la acción de saltar de alegría.

Caso de Uso Bailar

Actores Usuario

Tipo Primario

Descripción El usuario indica al sistema que quiere que su avatar baile. El sistema ordena al avatar que representa al usuario que realice la acción de bailar.

Caso de Uso Patalear

Actores Usuario

Tipo Primario

Descripción El usuario indica al sistema que quiere que su avatar patalee. El sistema ordena al avatar que representa al usuario que realice la acción de patalear.

Caso de Uso Comunicar avatar visto moviéndose

Actores Usuario

Tipo Primario

Descripción El usuario que se la liga, cuando su avatar se ha dado la vuelta después de contar, ve a algún jugador moviéndose y le indica al sistema que le diga a ese jugador que vuelva a la línea de inicio. El sistema le presenta al jugador un mensaje diciéndole que vuelva a la línea de partida.

Caso de Uso Volver a la línea de comienzo

Actores Usuario

Tipo Primario

Descripción Cuando un jugador gane la partida, el resto de sus jugadores le indicarán al sistema que coloque su avatar en la línea de comienzo. El sistema colocará al avatar en la línea de comienzo, mirando hacia donde está el jugador que se la liga.

Caso de Uso Andar

Actores Usuario

Tipo Primario

Descripción El usuario indica al sistema que ordene a su avatar que ande hacia adelante o hacia atrás. El sistema da la orden al avatar, que empezará a moverse.

Caso de Uso Correr

Actores Usuario

Tipo Primario

Descripción El usuario indica al sistema que ordene a su avatar que corra. El sistema da la orden al avatar, que empezará a moverse.

Caso de Uso Girar

Actores Usuario

Tipo Primario

Descripción El usuario indica al sistema que ordene a su avatar que se gire un número de grados. El sistema da la orden al avatar, que girará el ángulo indicado.

Caso de Uso Hablar con otros usuarios

Actores Usuario

Tipo Primario

Descripción El usuario, a través de un chat, escribe mensajes a otros usuarios. El sistema recoge los mensajes que ha escrito un usuario y se los manda al resto.

2.3. CONCEPTOS DE USO

Como se ha mencionado anteriormente, no todas las funciones que realiza el sistema se producen a través de la interacción con el usuario, lo cual imposibilita que se puedan definir a través de los casos de uso. Esto sucede con las acciones que el usuario delega en el sistema o con las que se realizan de manera automática, razón por la cual se utilizará la representación en forma de conceptos de uso propuesta en [Sánchez99], de manera que se volverán a definir algunas acciones para contemplar el caso en que no dependan del control del usuario y se contemplarán otras nuevas que no se realizaban cuando el usuario tenía el control de todas las acciones del avatar. Por tanto, con la notación en forma de conceptos de uso se contemplarán acciones que no son demandadas por el usuario.

CONCEPTO DE USO	CONCEPTO DE OPERACIÓN
Detectar avatar conocido	**Propósito:** comprobar si hay avatares conocidos alrededor para poder saludarles. **Modo de funcionamiento:** bien al conectarse a la partida, o posteriormente, detectar avatares conocidos. Para poder realizar esta actividad es necesario que el avatar tenga en algún sitio almacenadas aquellas amistades que ha ido haciendo en las diferentes conexiones al EV, en forma de una memoria del avatar. **Dinámica:** funciona cíclicamente.

CONCEPTO DE USO	CONCEPTO DE OPERACIÓN
Saludar a un avatar	**Propósito:** que los avatares se puedan hacer una señal de saludo con el fin de iniciar una conversación. **Modo de funcionamiento:** se trata de una actividad que se intentará automatizar, de modo que el avatar ha de tener memoria dentro del juego, de a quién desea saludar el usuario y a quién no. Además, dependerá de la personalidad del avatar el que éste salude o no y cómo lo haga. Podrá ser simplemente un leve movimiento de cabeza o agitar un brazo en el aire, estrechar las manos o abrazarse. **Dinámica:** el avatar sólo saludará cuando encuentre a un avatar conocido y al que desea emitir un saludo.

CONCEPTO DE USO	CONCEPTO DE OPERACIÓN
Detectar saludo	**Propósito:** que los avatares puedan detectar que otro avatar les está saludando. **Modo de funcionamiento:** se trata de una actividad asociada a la percepción del mundo exterior, y la reacción dependerá de si el avatar quiere o no responder al saludo que ha detectado. La detección del saludo ha de ser automática. Cada avatar será capaz de percibir el entorno, de modo que pueda saber si a una determinada distancia, que será su campo de visión, hay alguien intentando saludarle. **Dinámica:** esta actividad está constantemente activa, al igual que todas las actividades de percepción. Pero el resultado o la consecuencia de dicha percepción sólo tiene lugar cuando el avatar lo considera oportuno.

CONCEPTO DE USO	CONCEPTO DE OPERACIÓN
Hacer reír	**Propósito:** animar a otros avatares o distraerlos del juego. **Modo de funcionamiento:** un avatar simpático puede hacer reír a otro cuando detecta que éste último está triste, aburrido, etc. Incluso durante el juego, se puede utilizar como treta para distraer a los contrincantes. Se trata de una actividad automática. Si el avatar está cerca de otro triste, le hará gestos para que se ría. **Dinámica:** sólo son detectados los avatares tristes cuando están en el campo de visión de nuestro avatar o en el campo auditivo, en el caso en que estén llorando. Sólo podrá hacer reír a avatares que estén próximos, en su radio de acción.

CONCEPTO DE USO	CONCEPTO DE OPERACIÓN
Detectar que me hacen reír	**Propósito:** cambiar de humor, independientemente de si el avatar estaba normal, triste, alegre, etc. **Modo de funcionamiento:** si detecto un avatar en mi campo de visión que me hace reír, cambiaré mi humor en función del actual y de mi personalidad. **Dinámica:** funciona cíclicamente y se dispara sólo si el avatar que intenta hacerme reír está en el campo de visión de mi avatar.

CONCEPTO DE USO	CONCEPTO DE OPERACIÓN
Ponerse triste: refleja un estado de ánimo relacionado con lo acontecido en el EV.	**Propósito:** reflejar un estado de ánimo. **Modo de funcionamiento:** la tristeza se representa por acciones como: llorar, andar despacio, llevar la cabeza caída. El avatar deberá contar con la estructura necesaria para almacenar este estado de ánimo, al tiempo que se lo comunica a los demás. **Dinámica:** cuando el conjunto de ocurrencias en el EV es tal que produce la tristeza de un avatar, el cerebro, que es el conductor de las sensaciones y emociones, ordena al resto del cuerpo ponerse triste, lo cual se refleja en la apariencia externa del avatar, además de en la interna.

CONCEPTO DE USO	CONCEPTO DE OPERACIÓN
***Detectar avatar triste*:** sirve para detectar una situación en la que se pueda ayudar a otro avatar a salir de su estado de tristeza.	**Propósito:** detectar a otro avatar que está triste dentro del EV, para posteriormente animarle, por ejemplo haciéndole reír. **Modo de funcionamiento:** si un avatar tiene en su campo de visión a un avatar que está triste, actuará de un modo u otro dependiendo de la personalidad y el humor del avatar. Para que un avatar pueda detectar que otro está triste es necesario que se almacene el algún sitio el estado en que se encuentra cada avatar. **Dinámica:** funciona cíclicamente

CONCEPTO DE USO	CONCEPTO DE OPERACIÓN
Enfadarse: refleja un estado de ánimo relacionado con lo acontecido en el EV.	**Propósito:** reflejar un estado de ánimo. **Modo de funcionamiento:** el enfado se representa por acciones como andar con pasos enérgicos e ir con los brazos cruzados. El avatar deberá contar con la estructura necesaria para almacenar este estado de ánimo, al tiempo que se lo comunica a los demás. **Dinámica:** cuando el conjunto de ocurrencias en el EV es tal que produce el enfado de un avatar, el conductor de las sensaciones y emociones de éste ordena al resto del cuerpo ponerse en posición de enfado, lo cual se refleja en la apariencia externa del avatar, además de en la interna.

CONCEPTO DE USO	CONCEPTO DE OPERACIÓN
***Detectar avatar enfadado*:** sirve para detectar una situación en la que se pueda ayudar a otro avatar a salir de su estado de enfado o, por el contrario, alejarse por miedo a una reacción negativa por parte de éste.	**Propósito:** detectar a otro avatar que está enfadado dentro del EV, lo cual posiblemente desencadene una acción para consolarle. La acción que se realice depende básicamente de los rasgos de humor y personalidad del avatar. **Modo de funcionamiento:** si un avatar tiene en su campo de visión a un avatar que está enfadado, actuará de un modo u otro dependiendo de la personalidad y el humor del avatar. **Dinámica:** funciona cíclicamente.

CONCEPTO DE USO	CONCEPTO DE OPERACIÓN
***Aburrirse*:** refleja un estado de ánimo relacionado con lo acontecido en el EV.	**Propósito:** reflejar un estado de ánimo. **Modo de funcionamiento:** el aburrimiento se representa por acciones como: andar arrastrando los pies e ir con los brazos caídos. El avatar deberá contar con la estructura necesaria para almacenar este estado de ánimo, al tiempo que se lo comunica a los demás. **Dinámica:** cuando el conjunto de ocurrencias en el EV es tal que produce el aburrimiento de un avatar, conductor de las sensaciones y emociones de éste ordena al resto del cuerpo ponerse en posición de aburrimiento, lo cual se refleja en la apariencia externa del avatar, además de en la interna.

CONCEPTO DE USO	CONCEPTO DE OPERACIÓN
Detectar avatar aburrido: sirve para detectar una situación en la que se pueda ayudar a otro avatar a salir de su estado de aburrimiento.	**Propósito:** detectar a otro avatar que está aburrido, dentro del EV. **Modo de funcionamiento:** si un avatar tiene en su campo de visión a un avatar que está aburrido, actuará de un modo u otro dependiendo de la personalidad y el humor del avatar. **Dinámica:** funciona cíclicamente.

CONCEPTO DE USO	CONCEPTO DE OPERACIÓN
Detectar gente alrededor: durante el transcurso de la estancia en el EV puede interesar saber cuántas personas tengo alrededor, tanto mientras se pasea por el EV, como cuando se está jugando.	**Propósito:** saber el número de personas a mi alrededor para posteriormente reaccionar si fuese necesario. **Modo de funcionamiento:** si se detecta gente alrededor, se reaccionará de una manera u otra dependiendo de la personalidad y el humor del avatar. Por ejemplo, si está nervioso mirará a su alrededor desconfiado. **Dinámica:** funciona cíclicamente.

CONCEPTO DE USO	CONCEPTO DE OPERACIÓN
Detectar avatar lejos de la pared: el propio avatar detecta que se encuentra muy lejos de la pared. Esto puede influir en su estado de ánimo si, por ejemplo, el juego está muy avanzado, o le han pillado muchas veces moviéndose (durante el juego, claro).	**Propósito:** saber la distancia a la que se encuentra un avatar de la pared. **Modo de funcionamiento:** cuando el avatar detecta que está muy lejos de la pared debe reaccionar de alguna manera, puesto que esta situación puede afectar a su estado de humor. La forma en que esto quede reflejado exteriormente depende de su personalidad. **Dinámica:** funciona cíclicamente, y teniendo en consideración cosas como el tiempo que lleva transcurrido el juego, etc., se toman decisiones sobre cómo actuar.

CONCEPTO DE USO	CONCEPTO DE OPERACIÓN
Detectar avatar cerca de la pared: el propio avatar detecta que se encuentra muy cerca de la pared. Esto puede influir en su estado de ánimo si, por ejemplo, está muy cerca de la pared, y otros avatares le pisan los talones.	**Propósito:** saber la distancia a la que se encuentra un avatar de la pared. **Modo de funcionamiento:** cuando el avatar detecta que está próximo a la pared debe reaccionar de alguna forma, puesto que esta situación puede afectar a su estado de humor. Por supuesto, la forma en que quede reflejado exteriormente dependerá de su personalidad. **Dinámica:** funciona cíclicamente, y teniendo en consideración cosas como el número de avatares alrededor, etc., se toman decisiones sobre cómo actuar.

CONCEPTO DE USO	CONCEPTO DE OPERACIÓN
Detectar otros avatares cerca de la pared: el avatar comprueba si le van a ganar, y esto influye en el estado de ánimo.	**Propósito:** ver si hay alguien que me va a ganar. **Modo de funcionamiento:** en función de si me van a ganar o no, mi humor variará de una forma u otra. **Dinámica:** una vez conectado a la partida, y mientras el usuario no se la ligue, funcionará cíclicamente.

CONCEPTO DE USO	CONCEPTO DE OPERACIÓN
***Detectar jugadores moviéndose*:** esta es una actividad propia del juego que se pretende desarrollar en este EV. Sólo es propia del jugador que se la liga.	**Propósito:** mandar a los jugadores moviéndose a la línea de comienzo del juego. **Modo de funcionamiento:** el avatar que se la liga, cuando termina de contar, observa a los jugadores y señala a los que se están moviendo. **Dinámica:** se ejecuta cada vez que el avatar que se la liga deja de contar.

CONCEPTO DE USO	CONCEPTO DE OPERACIÓN
***Detectar avatar visto moviéndose*:** si un avatar ha sido visto moviéndose ha de volver a la línea de partida.	**Propósito:** que un avatar que ha sido sorprendido moviéndose vuelva a la línea de origen. **Modo de funcionamiento:** el avatar que ha sido descubierto moviéndose vuelve a la línea de partida. **Dinámica:** si se trata de un avatar controlado por usuario y con cámara, el usuario se moverá al lugar en el que da comienzo el juego, y por tanto su avatar también. Si el usuario juega con teclado, al observar que le han visto moviéndose debe ordenar al avatar que vuelva al origen. Si se trata de un agente autónomo volverá a la línea de partida cuando detecte que le han visto moviéndose.

CONCEPTO DE USO	CONCEPTO DE OPERACIÓN
***Detectar demasiadas veces visto*:** si un avatar ha sido visto moviéndose dentro del juego, no tiene importancia en principio, pero si este número de veces crece, puede influir en su estado de ánimo.	**Propósito:** indicarle al cerebro que me han visto muchas veces, en el transcurso del juego. **Modo de funcionamiento:** cuando el avatar detecta que ha vuelto muchas veces a la línea de partida como resultado de haber sido visto moviéndose, debe comprobar si esta situación afecta a su estado de ánimo. **Dinámica:** funciona cíclicamente, y teniendo en consideración la personalidad del avatar, se decidirá cómo afecta al humor del mismo.

CONCEPTO DE USO	CONCEPTO DE OPERACIÓN
Detectar duración de la partida	**Propósito:** detectar si la partida está durando mucho. **Modo de funcionamiento:** si la partida está siendo demasiado larga, puede afectar al humor del avatar. El decidir si una partida es demasiado larga o no dependerá de la personalidad de cada avatar. **Dinámica:** funciona cíclicamente.

CONCEPTO DE USO	CONCEPTO DE OPERACIÓN
Detectar avatar gana el juego: cuando un avatar llega a la pared donde está el que se la liga sin ser visto, gana.	**Propósito:** saber que un avatar ha ganado. **Modo de funcionamiento:** tanto para el avatar ganador como para el que se la liga, implica cambiar de rol. Además, el avatar que ha ganado debe mandarle al cerebro la información de que es el ganador. **Dinámica:** funciona cíclicamente.

CONCEPTO DE USO	CONCEPTO DE OPERACIÓN
***Detectar avatar pierde el juego*:** cuando un avatar llega a la pared donde está el que se la liga sin ser visto, gana; el resto pierden el juego.	**Propósito:** hacer que el resto de avatares sepan que han perdido y vuelvan a la línea de origen. **Modo de funcionamiento:** cuando un avatar detecta que otro ha ganado el juego, cambiará o no su humor dependiendo de su personalidad. **Dinámica:** funciona cíclicamente.

CONCEPTO DE USO	CONCEPTO DE OPERACIÓN
Detectar obstáculo	**Propósito:** pararse antes de colisionar con un objeto del EV. Con esta funcionalidad, lo que se pretende es que el avatar sea capaz de detectar una colisión y no atraviese los objetos que están en el EV. **Modo de funcionamiento:** el detector de colisión sólo hará que el avatar se pare antes de colisionar con cualquier objeto del EV. Tras una colisión será el usuario el que indique a su avatar si debe ir a la derecha, izquierda o atrás. **Dinámica:** el detector de colisiones debe estar activo siempre que el avatar esté moviéndose por el EV.

CONCEPTO DE USO	CONCEPTO DE OPERACIÓN
Volverse hacia la pared para contar: esta es una actividad propia del juego que se va a realizar dentro del EV. Sólo es propia del jugador que en ese momento se la ligue.	**Propósito:** que el que se la liga se vuelva hacia la pared de modo que no vea a los usuarios y estos puedan avanzar hacia la pared en la que éste se encuentra. **Modo de funcionamiento:** el avatar que se la liga gira 180 grados para colocarse mirando a la pared. **Dinámica:** se ejecuta cada vez que el avatar que se la liga va a empezar a contar.

CONCEPTO DE USO	CONCEPTO DE OPERACIÓN
Detectar que el que se la liga está contando	**Propósito:** saber cuándo puedo avanzar hacia la pared sin ser visto. **Modo de funcionamiento:** cuando creo que el que se la liga no me ve, avanzo hacia la pared. **Dinámica:** funciona cíclicamente, una vez comenzada la partida y sólo si no soy el que se la liga.

CONCEPTO DE USO	CONCEPTO DE OPERACIÓN
Volverse hacia los jugadores: es una actividad propia del juego que se va a desarrollar dentro del EV. La realiza el jugador que en ese momento se la ligue.	**Propósito:** que el avatar que se la liga mire hacia los jugadores. **Modo de funcionamiento:** el avatar que se la liga se gira 180 grados. **Dinámica:** se ejecuta cada vez que el que se la liga deja de contar.

CONCEPTO DE USO	CONCEPTO DE OPERACIÓN
Decidir quién se la liga**:** a la hora de asignar roles a los usuarios del juego, es necesario decidir quién se la liga.	**Propósito:** saber si el avatar que se acaba de conectar toma el rol de participante o se la liga. **Modo de funcionamiento:** cada jugador, cuando se conecta al EV, sabe si había algún otro conectado; si lo había, él toma el rol de jugador. Si no había nadie, él toma el rol de avatar que se la liga. **Dinámica:** se elige sólo al principio del juego, el resto de las veces pasa a ligarla el que gana.

CONCEPTO DE USO	CONCEPTO DE OPERACIÓN
***Agredir*:** se puede decidir agredir a un avatar como resultado de un estado de ánimo muy enfadado.	**Propósito:** agredir a otro avatar. **Modo de funcionamiento:** si el avatar está muy enfadado, puede decidir agredir a otro avatar, dependiendo de su personalidad. **Dinámica:** se dispara bajo demanda, dependiendo de la personalidad y el humor del avatar.

CONCEPTO DE USO	CONCEPTO DE OPERACIÓN
Detectar agresión: como resultado de una agresión, el avatar caerá al suelo.	**Propósito:** mostrar físicamente el resultado de la agresión. **Modo de funcionamiento:** sólo tiene efecto cuando un avatar detecta que ha sido agredido. **Dinámica:** se dispara cada vez que un avatar sufre una agresión.

CONCEPTO DE USO	CONCEPTO DE OPERACIÓN
Volver a la línea de partida	**Propósito:** en el caso de que el avatar esté dirigido por un usuario, este procedimiento no es útil; en cambio sí lo es para el caso en que el avatar esté dirigido por un agente autónomo. **Modo de funcionamiento:** que un avatar camine hacia la línea de partida. **Dinámica:** bien como resultado de haber sido sorprendido moviéndose o bien como resultado de perder la partida.

CONCEPTO DE USO	CONCEPTO DE OPERACIÓN
Reír	**Propósito:** mostrar un estado de ánimo. **Modo de funcionamiento:** posible reacción ante una situación que produce la risa. **Dinámica:** cuando el humor y la personalidad del avatar indican que se puede hacer.

CONCEPTO DE USO	CONCEPTO DE OPERACIÓN
Saltar de Alegría	**Propósito:** mostrar un estado de ánimo. **Modo de funcionamiento:** posible reacción ante una situación de felicidad **Dinámica:** cuando el humor y la personalidad del avatar indican que se puede hacer.

CONCEPTO DE USO	CONCEPTO DE OPERACIÓN
Bailar	**Propósito:** mostrar un estado de ánimo. **Modo de funcionamiento:** posible reacción ante una situación de felicidad. **Dinámica:** cuando el humor y la personalidad del avatar indican que se puede hacer.

CONCEPTO DE USO	CONCEPTO DE OPERACIÓN
Patalear	**Propósito:** mostrar un estado de ánimo. **Modo de funcionamiento:** posible reacción ante la detección de una agresión o un enfado. **Dinámica:** cuando el humor y la personalidad del avatar indican que se puede hacer.

CONCEPTO DE USO	CONCEPTO DE OPERACIÓN
Llorar	**Propósito:** mostrar un estado de ánimo. **Modo de funcionamiento:** posible reacción ante una situación de tristeza o ante la detección de una agresión. **Dinámica:** cuando el humor y la personalidad del avatar indican que se puede hacer.

2.4. PRIORIZACIÓN DE CASOS Y CONCEPTOS DE USO

Para realizar esta actividad nos hemos visto obligados a tomar en consideración, como ya se ha visto, elementos que no están utilizados por Larman y que ni siquiera se consideran dentro de UML, los conceptos de uso, pero sin los cuales no es posible especificar el sistema en su totalidad. Es por ello por lo que a partir de este momento habrá que contar con ellos para, de alguna forma, añadírselos al método de Larman y poder continuar con el desarrollo.

El primer paso, por tanto, es incluirlos junto con los casos de uso para poder realizar una priorización y asignarlos a sucesivos ciclos de desarrollo.

El criterio que se va a seguir para realizar la priorización va a venir dado por la información que se posee para desarrollar los casos de uso y los conceptos de uso que se han descrito anteriormente. Concretamente, puesto que aún no se sabe cómo pueden encajar los conceptos de uso en el método de Larman y muchos de ellos se refieren a funcionalidades descritas en casos de uso que el usuario puede delegar en el sistema, la decisión que se ha tomado ha sido desarrollar en primer lugar las funciones contenidas en los casos de uso para, posteriormente, con la información que de ahí se pueda sacar, desarrollar los conceptos de uso. Con esto, la asignación que se obtiene para los ciclos de desarrollo es la que se muestra a continuación:

- Primer Ciclo:
  - Conectarse al Mundo Virtual
  - Contar
  - Comunicar avatar visto moviéndose
  - Volver a la línea de comienzo
  - Andar
  - Correr
  - Girar
  - Saludar a otro avatar
  - Reír
  - Hacer reír a otro avatar
  - Agredir a otro avatar
  - Llorar
  - Saltar de alegría
  - Bailar
  - Patalear
  - Hablar con otros usuarios

- Segundo Ciclo:
  - Detectar avatar conocido
  - Saludar a un avatar
  - Detectar saludo
  - Hacer reír
  - Detectar que me hacen reír
  - Reír
  - Saltar de Alegría
  - Bailar
  - Ponerse triste
  - Detectar avatar triste
  - Llorar
  - Enfadarse
  - Detectar avatar enfadado
  - Patalear
  - Agredir
  - Detectar agresión
  - Aburrirse
  - Detectar avatar aburrido
  - Detectar gente alrededor
  - Detectar avatar lejos de la pared
  - Detectar avatar cerca de la pared
  - Detectar otros avatares cerca de la pared
  - Detectar jugadores moviéndose
  - Detectar avatar visto moviéndose
  - Volver a la línea de partida

- Detectar demasiadas veces visto
- Detectar duración de la partida
- Detectar avatar gana el juego
- Detectar avatar pierde el juego
- Volverse hacia la pared para contar
- Detectar que el que se la liga está contando
- Volverse hacia los jugadores
- Decidir quién se la liga
- Buscar puerta
- Detectar obstáculo

3. GLOSARIO

A continuación se presenta un glosario de los términos utilizados en las fases de Especificación de Requisitos y Análisis y Diseño de los dos ciclos de desarrollo.

Término	Categoría	Descripción
Aburrirse	*Caso de Uso*	Procedimiento para que un usuario pueda hacer que su avatar se aburra.
Aburrirse	*Concepto de Uso*	Procedimiento para que un avatar pueda decidir que se aburre.
Agredir	*Concepto de Uso*	Mecanismo para que un avatar decida si quiere agredir a otro.
Agredir a otro avatar	*Caso de Uso*	Muestra la manera en que un avatar controlado por el usuario puede agredir a otro.
Andar	*Caso de Uso*	Procedimiento para que un usuario pueda hacer que su avatar ande.
Bailar	*Caso de Uso*	Procedimiento para que un usuario pueda hacer que su avatar baile.
Bailar	*Concepto de Uso*	Mecanismo con el cual un avatar decide cuándo quiere bailar.
Buscar puerta	*Concepto de Uso*	Mecanismo para que un avatar pueda buscar una puerta para abandonar el EV.
Comunicar avatar visto moviéndose	*Caso de Uso*	Muestra el procedimiento que se sigue para que el avatar que maneja el usuario que se la liga pueda comunicar a otro que ha sido visto moviéndose.
Conectarse al Mundo Virtual	*Caso de Uso*	Muestra el procedimiento a seguir para que un usuario pueda conectarse al EV.
Contar	*Caso de Uso*	Muestra el procedimiento que se sigue para que el avatar que maneja el usuario que se la liga se dé la vuelta hacia la pared y cuente, para que el resto pueda avanzar.
Correr	*Caso de Uso*	Procedimiento para que un usuario pueda hacer que su avatar corra.
Decidir quién se la liga	*Concepto de Uso*	Mecanismo para que los avatares decidan quién se la liga.
Detectar agresión	*Concepto de Uso*	Mecanismo para que un avatar detecte que le han agredido.
Detectar avatar aburrido	*Concepto de Uso*	Mecanismo para que un avatar decida si se aburre.
Detectar avatar cerca de la pared	*Concepto de Uso*	Mecanismo para que un avatar vea si está cerca de la pared.
Detectar avatar conocido	*Concepto de Uso*	Mecanismo para que un avatar detecte a otros avatares a los que conoce.
Detectar avatar enfadado	*Concepto de Uso*	Mecanismo para que un avatar compruebe si otro avatar está enfadado.

Detectar avatar gana el juego	*Concepto de Uso*	Mecanismo para que un avatar sepa si ha ganado una partida.
Detectar avatar lejos de la pared	*Concepto de Uso*	Mecanismo para que un avatar vea si está lejos de la pared.
Detectar avatar pierde el juego	*Concepto de Uso*	Mecanismo para que un avatar sepa si ha perdido una partida.
Detectar avatar triste	*Concepto de Uso*	Mecanismo para que un avatar compruebe si otro avatar está triste.
Detectar avatar visto moviéndose	*Concepto de Uso*	Mecanismo para que el avatar que se la liga compruebe si hay otros avatares moviéndose.
Detectar demasiadas veces visto	*Concepto de Uso*	Mecanismo para que el avatar decida si el jugador que se la liga le ha visto moverse muchas veces.
Detectar duración de la partida	*Concepto de Uso*	Mecanismo para que el avatar compruebe cuánto está durando una partida.
Detectar gente alrededor	*Concepto de Uso*	Mecanismo para que un avatar detecte si tiene otros avatares alrededor.
Detectar jugadores moviéndose	*Concepto de Uso*	Mecanismo para que el avatar que se la liga sepa si hay jugadores moviéndose.
Detectar obstáculo	*Concepto de Uso*	Mecanismo para que un avatar detecte si hay un obstáculo que no le deja moverse.
Detectar otros avatares cerca de la pared	*Concepto de Uso*	Mecanismo para que un avatar pueda saber si hay otros avatares cerca de la pared.
Detectar que el que se la liga está contando	*Concepto de Uso*	Mecanismo para que un avatar pueda saber si el que se la liga está contando y pueda decidir si se mueve o no.
Detectar que me hacen reír	*Concepto de Uso*	Mecanismo para que un avatar sepa si hay otro avatar intentando hacerle reír.
Detectar saludo	*Concepto de Uso*	Mecanismo para que un avatar sepa si hay algún avatar saludándolo.
Enfadarse	*Caso de Uso*	Procedimiento para que un usuario pueda hacer que su avatar se enfade.
Enfadarse	*Concepto de Uso*	Mecanismo utilizado para que un avatar pueda mostrar un estado de humor correspondiente al enfado.
Girar	*Caso de Uso*	Procedimiento para que un usuario pueda hacer que su avatar gire.
Hablar con otros usuarios	*Caso de Uso*	Procedimiento para que un usuario pueda comunicarse con otros usuarios a través de un chat.
Hacer reír	*Concepto de Uso*	Procedimiento para que un avatar pueda hacer reír a otro.
Hacer reír a otro avatar	*Caso de Uso*	Muestra la manera en que un avatar controlado por el usuario puede hacer reír a otro.
Llorar	*Caso de Uso*	Procedimiento para que un usuario pueda hacer que su avatar llore.

Llorar	*Concepto de Uso*	Procedimiento para que un avatar exprese un alto grado de tristeza.
Patalear	*Caso de Uso*	Procedimiento para que un usuario pueda hacer que su avatar patalee.
Patalear	*Concepto de Uso*	Procedimiento para que un avatar pueda expresar su enfado.
Ponerse triste	*Caso de Uso*	Procedimiento para que un usuario pueda hacer que su avatar se ponga triste.
Ponerse triste	*Concepto de Uso*	Mecanismo para que un avatar pueda mostrar su grado de tristeza.
Reír	*Caso de Uso*	Procedimiento para que un usuario pueda hacer que su avatar se ría.
Reír	*Concepto de Uso*	Procedimiento para que un avatar exprese su grado de alegría.
Saltar de alegría	*Caso de Uso*	Procedimiento para que un usuario pueda hacer que su avatar salte de alegría.
Saltar de Alegría	*Concepto de Uso*	Procedimiento para que un avatar exprese su grado de alegría.
Saludar a otro avatar	*Caso de Uso*	Muestra el procedimiento para que un avatar controlado por el usuario pueda saludar a otro.
Saludar a un avatar	*Concepto de Uso*	Procedimiento para que un avatar pueda saludar a otro.
Volver a la línea de comienzo	*Caso de Uso*	Procedimiento para que un usuario pueda hacer que su avatar vuelva a la línea de comienzo.
Volver a la línea de partida	*Concepto de Uso*	Mecanismo para que un avatar vuelva a la línea de partida.
Volverse hacia la pared para contar	*Concepto de Uso*	Mecanismo para que el jugador que se la liga decida cuándo darse la vuelta para contar.
Volverse hacia los jugadores	*Concepto de Uso*	Mecanismo para que el avatar que se la liga decida cuándo ha terminado de contar y se dé la vuelta para mirar a los otros jugadores.

PRIMER CICLO DE DESARROLLO

1. ANÁLISIS

Como se ha mencionado anteriormente, en este primer ciclo se van a desarrollar todas aquellas funcionalidades que se han podido expresar en forma de casos de uso.

Se comenzará con la descripción de los casos de uso en formato extendido, lo que permitirá tener una visión más exacta de qué y cómo se deberá realizar lo descrito en los casos de uso en formato de alto nivel.

A partir de esta descripción, se realizará la construcción de los diagramas de secuencia del sistema, de los cuales se obtendrán uno por cada curso típico de eventos de los casos de uso y otro por cada curso alternativo.

Posteriormente, con la información que se pueda extraer de lo realizado hasta ese momento, se realizará la construcción del modelo conceptual, para dar una primera visión de lo que será el sistema.

Por último, para terminar con la fase de análisis, se describirá, por cada operación aparecida en los diagramas de secuencia del sistema, un contrato de operación, donde se dejará claro qué debe hacer cada una de estas operaciones.

1.1. CASOS DE USO EN FORMATO EXPANDIDO

Caso de Uso	Conectarse al Mundo Virtual
Actores	Usuario (iniciador)
Propósito	Iniciar una sesión en el EV
Tipo	Primario y esencial
Visión General	El usuario inicia la ejecución del EV. El sistema le pide que seleccione un avatar, un punto de vista, los rasgos de personalidad de su avatar, un servidor de conexión al EV y un dispositivo de captura de movimiento. El usuario introduce todos los datos pedidos y el sistema realiza la conexión al EV, asignándole un rol al jugador.
Referencias	Req(01), Req(17), Req(38), Req(39), Req(40), Req(41)

Curso típico de eventos:

Usuario	Sistema
1. Selecciona un avatar.	2. Anota la selección y pide que se seleccione el punto de vista que se usará.
3. Selecciona el punto de vista.	4. Anota la selección de punto de vista y pide los rasgos de personalidad del avatar.
5. Elige los rasgos de personalidad del avatar.	6. Anota los rasgos de personalidad y pide el servidor de simulación al que se conectará el usuario.
7. Selecciona el servidor de conexión.	8. Anota el servidor de conexión y pide el dispositivo de captura de movimiento.
9. Elige el dispositivo de captura de movimiento.	10. Inicializa el dispositivo de captura de movimiento, se conecta al servidor de mundos virtuales e inicializa el entorno.

Cursos alternativos:

- Paso 10: No se puede inicializar el dispositivo de captura de movimiento. Se da un mensaje de error y se finaliza la ejecución.
- Paso 10: No se puede establecer la conexión con el servidor. Se da un mensaje de error y se finaliza la ejecución.
- Paso 10: Falla la inicialización. Se da un mensaje de error y termina la ejecución.

Caso de Uso Contar

Actores Usuario (iniciador)

Propósito Que el jugador que se la liga se vuelva a la pared para contar.

Tipo Primario y esencial

Visión General El jugador que se la liga indica al sistema que quiere empezar a contar. El sistema indica al avatar que se dé la vuelta hacia la pared; tras esto, indicará al avatar que cuente y, por último, que se dé la vuelta hacia los jugadores.

Referencias Req(02), Req(04)

Curso típico de eventos:

Usuario	Sistema
1. Selecciona la acción contar.	2. Comprueba que el avatar no está realizando ninguna otra acción y le ordena que realice la acción de contar.

Cursos alternativos:

- Paso 2: Se está realizando otra acción. No se lleva a cabo la acción de contar.

Caso de Uso Saludar a otro avatar

Actores Usuario (iniciador)

Propósito Que un avatar pueda saludar a otros.

Tipo Primario y esencial

Visión General El usuario indica al sistema que quiere saludar, y éste le indica al avatar que representa a este usuario que realice la acción de saludar que esté de acuerdo con su personalidad y su humor.

Referencias Req(20)

Curso típico de eventos:

Usuario	Sistema
1. Selecciona la acción de saludar.	2. Comprueba que el avatar no esté realizando ninguna otra acción y le pide al usuario que seleccione el avatar al que quiere saludar.
3. Selecciona el avatar al que quiere saludar.	4. Ordena al avatar del usuario que salude al avatar elegido, de acuerdo con su personalidad y su humor.

Cursos alternativos:

- Paso 2: El avatar está realizando alguna otra acción. No se ejecuta la acción de saludar.

Caso de Uso Hacer reír a otro avatar

Actores Usuario

Propósito Que un avatar pueda hacer reír a otro para cambiar su estado de ánimo o para distraerle.

Tipo Primario y esencial

Visión General El usuario indica al sistema que quiere hacer reír a otro avatar. El sistema le indica al avatar que representa a este usuario que realice la acción de hacer reír que esté de acuerdo con su personalidad y su humor.

Referencias Req(22)

Curso típico de eventos:

Usuario	Sistema
1. Selecciona la acción de hacer reír a otro avatar.	2. Comprueba que el avatar no está realizando ninguna otra acción y le solicita al usuario que elija el avatar al que quiere hacer reír.
3. Selecciona el avatar al que quiere hacer reír.	4. Ordena al avatar del usuario que haga reír al avatar elegido, de acuerdo con su personalidad y su humor.

Cursos alternativos:

- Paso 2: El avatar está realizando otra acción. No se realiza la acción de hacer reír a otro avatar.

Caso de Uso Agredir a otro avatar

Actores Usuario (iniciador)

Propósito Que un avatar agreda a otro, para distraerlo o para cambiar su humor.

Tipo Primario y esencial

Visión General El usuario indica al sistema que quiere agredir a otro avatar. El sistema le indica al avatar que representa a este usuario que realice la acción de agredir que esté de acuerdo con su personalidad y su humor.

Referencias Req(31)

Curso típico de eventos:

Usuario	Sistema
1. Selecciona la acción de agredir a otro avatar.	2. Comprueba que el avatar no está realizando ninguna otra acción y solicita al usuario que elija el avatar al que quiere agredir.
3. Selecciona el avatar al que quiere agredir.	4. Ordena al avatar del usuario que agreda al avatar elegido, de acuerdo con su personalidad y su humor.

Cursos alternativos:

- Paso 2: El avatar está realizando otra acción. No se realiza la acción de agredir a otro avatar.

Caso de Uso Reír

Actores Usuario

Propósito Que un avatar se pueda reír en respuesta a algo que sucede en el EV.

Tipo Primario y esencial

Visión General El usuario indica al sistema que quiere que su avatar se ría. El sistema ordena al avatar que representa al usuario que realice la acción de reír que esté de acuerdo con su humor y su personalidad.

Referencias Req(33)

Curso típico de eventos:

Usuario	Sistema
1. Selecciona la acción de reír.	2. Comprueba que el avatar no está realizando ninguna otra acción y ejecuta la acción de reír que esté de acuerdo con su estado de ánimo y sus rasgos de personalidad.

Cursos alternativos:

- Paso 2: El avatar está realizando otra acción, por lo que no se ejecuta la acción de reír.

Caso de Uso Llorar

Actores Usuario (iniciador)

Propósito Que un avatar pueda mostrar un alto grado de tristeza, en respuesta a algo que sucede en el EV.

Tipo Primario y esencial

Visión General El usuario indica al sistema que quiere que su avatar llore. El sistema ordena al avatar que representa al usuario que realice la acción de llorar que esté de acuerdo con su humor y su personalidad.

Referencias Req(37)

Curso típico de eventos:

Usuario	Sistema
1. Selecciona la acción de llorar.	2. Comprueba que el avatar no esté realizando ninguna otra acción y ejecuta la acción de llorar que esté de acuerdo con la personalidad y el estado de ánimo del mismo.

Cursos alternativos:

- Paso 2: Se está realizando otra acción. No se realiza la acción de llorar.

Caso de Uso Ponerse triste

Actores Usuario

Propósito Que un avatar pueda mostrar su estado de tristeza.

Tipo Primario y esencial

Visión General El usuario indica al sistema que quiere que su avatar se ponga triste. El sistema ordena al avatar que representa al usuario que realice la acción de ponerse triste que esté de acuerdo con su humor y su personalidad.

Referencias Req(24)

Curso típico de eventos:

Usuario	Sistema
1. Selecciona la acción de ponerse triste.	2. Comprueba que el avatar no esté realizando ninguna otra acción y ejecuta la acción de ponerse triste que esté de acuerdo con la personalidad y el estado de ánimo del mismo.

Cursos alternativos:

- Paso 2: Se está realizando otra acción. No se realiza la acción de ponerse triste.

Caso de Uso Enfadarse

Actores Usuario

Propósito Que un avatar pueda mostrar su estado de enfado.

Tipo Primario y esencial

Visión General El usuario indica al sistema que quiere que su avatar se enfade. El sistema ordena al avatar que representa al usuario que realice la acción de enfadarse que esté de acuerdo con su humor y su personalidad.

Referencias Req(26)

Curso típico de eventos:

Usuario	Sistema
1. Selecciona la acción de enfadarse.	2. Comprueba que el avatar no esté realizando ninguna otra acción y ejecuta la acción de enfadarse que esté de acuerdo con la personalidad y el estado de ánimo del mismo.

Cursos alternativos:

- Paso 2: Se está realizando otra acción. No se realiza la acción de enfadarse.

Caso de Uso Aburrirse

Actores Usuario

Propósito Que un avatar pueda exteriorizar su grado de aburrimiento.

Tipo Primario y esencial

Visión General El usuario indica al sistema que quiere que su avatar adopte la posición de aburrirse. El sistema ordena al avatar del usuario que realice la acción de aburrirse que esté de acuerdo con su humor y su personalidad.

Referencias Req(28)

Curso típico de eventos:

Usuario	Sistema
1. Selecciona la acción de aburrirse.	2. Comprueba que el avatar no esté realizando ninguna otra acción y ejecuta la acción de aburrirse que esté de acuerdo con la personalidad y el estado de ánimo del mismo.

Cursos alternativos:

- Paso 2: Se está realizando otra acción. No se realiza la acción de llorar.

Caso de Uso Saltar de alegría

Actores Usuario (iniciador)

Propósito Que un avatar pueda exteriorizar un estado de ánimo muy alegre.

Tipo Primario y esencial

Visión General El usuario indica al sistema que quiere que su avatar salte de alegría. El sistema ordena al avatar que representa al usuario que realice la acción de saltar de alegría.

Referencias Req(34)

Curso típico de eventos:

Usuario	Sistema
1. Selecciona la acción de saltar de alegría.	2. Comprueba que el avatar no esté ejecutando ninguna otra acción y ejecuta la acción de saltar de alegría, de acuerdo con la personalidad y el humor del mismo.

Cursos alternativos:

- Paso 2: Se está realizando otra acción. No se ejecuta la acción de saltar de alegría.

Caso de Uso Bailar

Actores Usuario (iniciador)

Propósito Que un avatar pueda mostrar un estado de ánimo alegre.

Tipo Primario y esencial

Visión General El usuario indica al sistema que quiere que su avatar baile. El sistema ordena al avatar que representa al usuario que realice la acción de bailar.

Referencias Req(35)

Curso típico de eventos:

Usuario	Sistema
1. Selecciona la acción de bailar.	2. Comprueba que el avatar no está ejecutando ninguna otra acción y ejecuta la acción de bailar que esté de acuerdo con sus rasgos de personalidad y su humor.

Cursos alternativos:

- Paso 2: El avatar está realizando otra acción y no se ejecuta la acción de bailar.

Caso de Uso Patalear

Actores Usuario

Propósito Que el avatar pueda mostrar un alto grado de enfado.

Tipo Primario y esencial

Visión General El usuario indica al sistema que quiere que su avatar patalee. El sistema ordena al avatar que representa al usuario que realice la acción de patalear.

Referencias Req(36)

Curso típico de eventos:

Usuario	Sistema
1. Selecciona la acción de patalear.	2. Comprueba que el avatar no está realizando ninguna otra acción y ejecuta la acción de patalear, de acuerdo con el estado de ánimo y la personalidad del avatar.

Cursos alternativos:

- Paso 2: El avatar está realizando otra acción y no se ejecuta la acción de patalear.

Caso de Uso Comunicar avatar visto moviéndose

Actores Usuario (iniciador)

Propósito Que un avatar sea informado de que el jugador que se la liga lo ha visto moviéndose.

Tipo Primario y esencial

Visión General El usuario que se la liga, cuando su avatar se ha dado la vuelta después de contar, ve a algún jugador moviéndose y le indica al sistema que le diga a ese jugador que vuelva a la línea de inicio. El sistema le presenta al jugador un mensaje diciéndole que vuelva a la línea de partida.

Referencias Req(8)

Curso típico de eventos:

Usuario	Sistema
1. Elige la acción de comunicar a otro usuario que lo ha visto moviéndose.	2. Comprueba que el usuario es el que se la liga y pregunta a qué usuario se le quiere decir que lo ha visto moverse.
3. Selecciona el avatar al que ha visto moviéndose.	4. Manda un mensaje al usuario del avatar seleccionado, comunicándole que ha sido visto moviéndose.

Cursos alternativos:

- Paso 2: El usuario no se la liga y se cancela la acción.

Caso de Uso Volver a la línea de comienzo

Actores Usuario (iniciador)

Propósito Volver a la línea de comienzo cuando otro jugador gana la partida o cuando nos han visto moviéndonos.

Tipo Primario y esencial

Visión General Cuando un jugador gane la partida, el resto de sus jugadores le indicarán al sistema que coloque su avatar en la línea de comienzo. El sistema colocará al avatar en la línea de comienzo, mirando hacia donde está el jugador que se la liga.

Referencias Req(14)

Curso típico de eventos:

Usuario	Sistema
1. Selecciona la acción de volver a la línea de comienzo.	2. Comprueba que el avatar no está realizando ninguna otra acción y le ordena que vuelva a la línea de comienzo.

Cursos alternativos:

- Paso 2: El avatar está realizando otra acción y se cancela la vuelta a la línea de comienzo.

Caso de Uso Andar

Actores Usuario (iniciador)

Propósito Que el avatar se pueda desplazar por el EV.

Tipo Primario y esencial

Visión General El usuario indica al sistema que ordene a su avatar que ande hacia adelante o hacia atrás. El sistema da la orden al avatar, que empezará a moverse.

Referencias Req(16)

Curso típico de eventos:

Usuario	Sistema
1. Selecciona la acción de andar.	2. Comprueba que el avatar esté parado o corriendo y que no esté realizando ninguna otra acción, y le da la orden de andar.

Cursos alternativos:

- Paso 2: El avatar no está parado o corriendo o está realizando alguna otra acción; se cancela la ejecución de la orden de andar.

Caso de Uso Correr

Actores Usuario (iniciador)

Propósito Que el avatar se pueda desplazar por el EV.

Tipo Primario y esencial

Visión General El usuario indica al sistema que ordene a su avatar que corra. El sistema da la orden al avatar, que empezará a moverse.

Referencias Req(15)

Curso típico de eventos:

Usuario	Sistema
1. Selecciona la acción de correr.	2. Comprueba que el avatar esté parado o andando y que no esté realizando ninguna otra acción, y le da la orden de correr.

Cursos alternativos:

- Paso 2: El avatar no está parado o andando o está realizando alguna otra acción; se cancela la ejecución de la orden de correr.

Caso de Uso Girar

Actores Usuario (iniciador)

Propósito Que el avatar pueda moverse con libertad por el EV.

Tipo Primario y esencial

Visión General El usuario indica al sistema que ordene a su avatar que se gire un número de grados. El sistema da la orden al avatar, que girará el ángulo indicado.

Referencias Req(43)

Curso típico de eventos:

Usuario	Sistema
1. Selecciona la acción de girar.	2. Comprueba que el avatar está parado, andando o corriendo y no está ejecutando ninguna otra acción, y le ordena que realice la acción de girar.

Cursos alternativos:

- Paso 2:El avatar está realizando alguna otra acción, y se cancela la acción de girar.

Caso de Uso Hablar con otros usuarios

Actores Usuario (iniciador)

Propósito Que los usuarios se puedan comunicar entre sí mientras estén conectados al EV.

Tipo Primario y esencial

Visión General El usuario, a través de un chat, escribe mensajes a otros usuarios. El sistema recoge los mensajes que ha escrito un usuario y se los manda al resto.

Referencias Req(47)

Curso típico de eventos:

Usuario	Sistema
1. Escribe un texto en la línea de escritura del chat y selecciona la opción de enviar el mensaje.	2. Comprueba que el mensaje no esté vacío y se lo manda al resto de los usuarios.

Cursos alternativos:

- No existen.

1.2. DIAGRAMAS DE SECUENCIA DEL SISTEMA

1.2.1. Caso de Uso *Conectarse al Mundo Virtual*

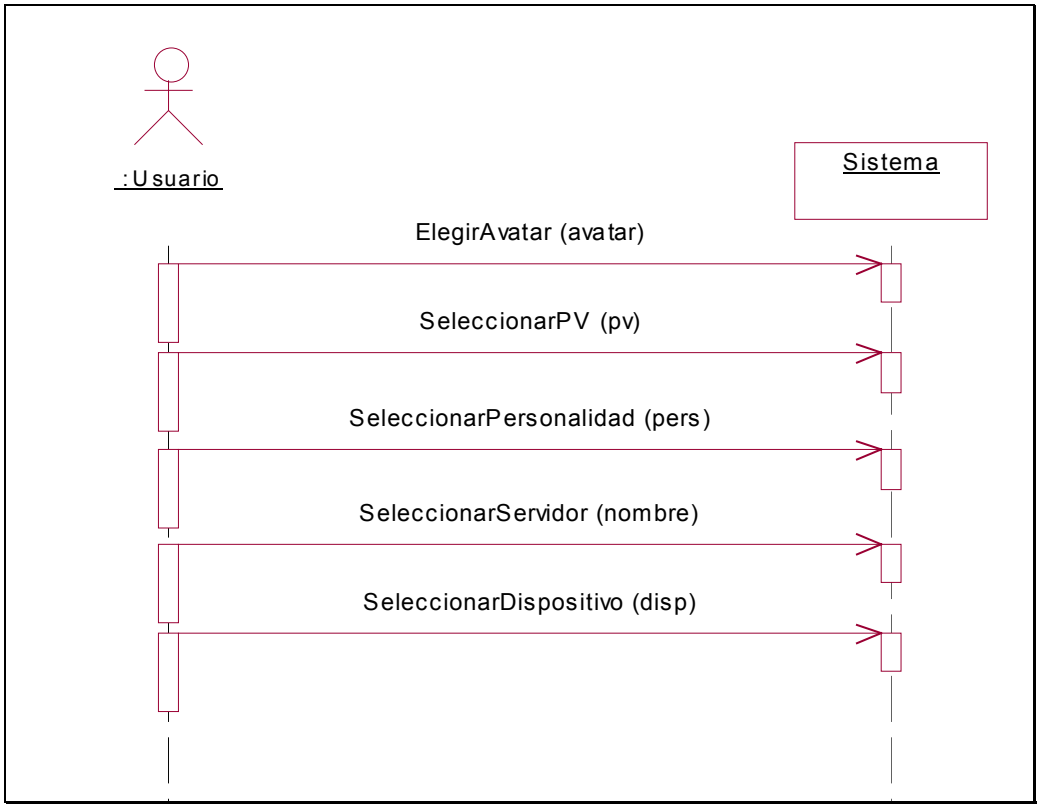

Figura 8: Conectarse al Mundo Virtual

1.2.2. Caso de Uso *Contar*

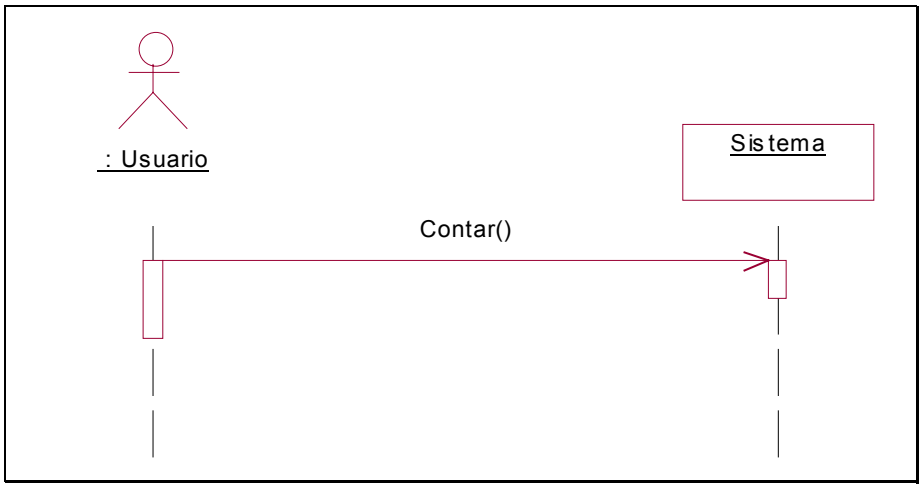

Figura 9: Contar

1.2.3. Caso de Uso *Saludar a otro Avatar*

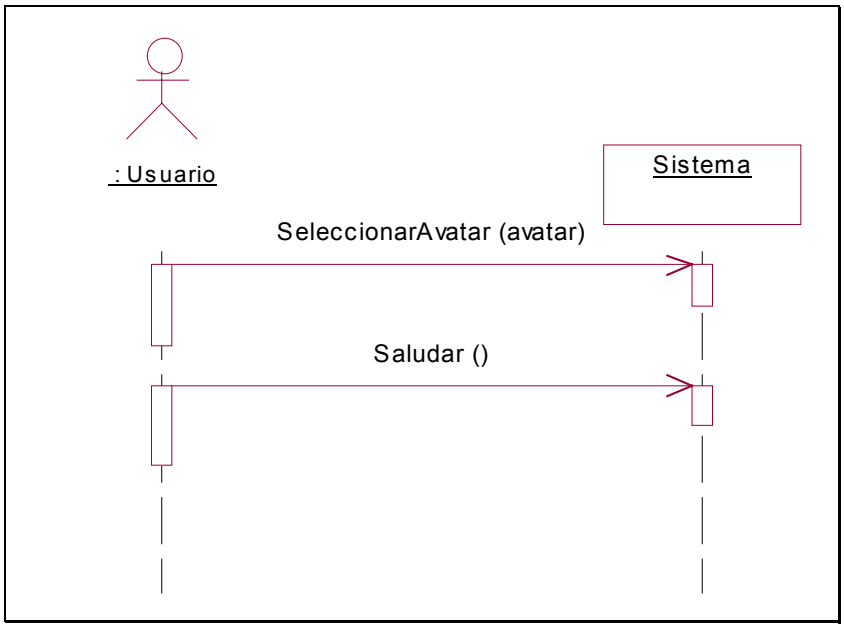

Figura 10: Saludar a otro Avatar

1.2.4. Caso de Uso *Hacer Reír a otro Avatar*

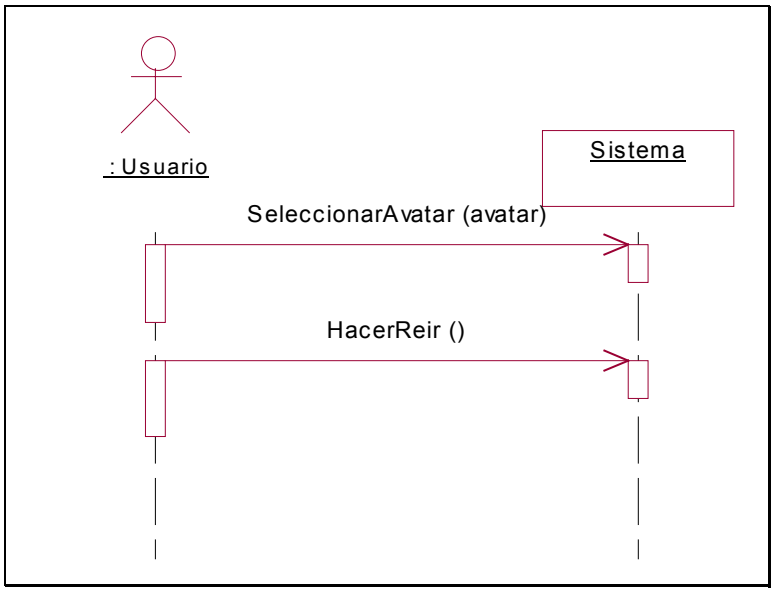

Figura 11: Hacer Reír a otro Avatar

1.2.5. Caso de Uso *Agredir a otro Avatar*

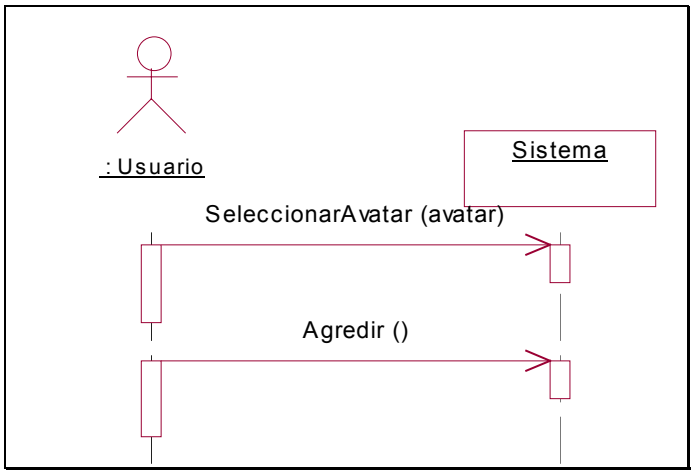

Figura 12: Agredir a otro Avatar

1.2.6. Caso de Uso *Reír*

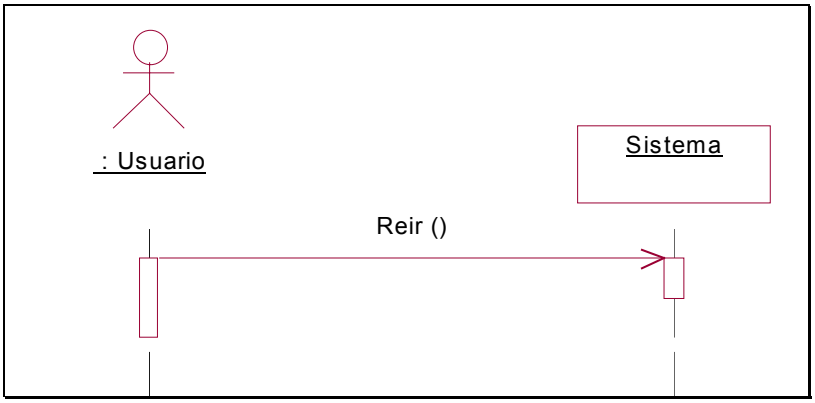

Figura 13: Reír

1.2.7. Caso de Uso *Llorar*

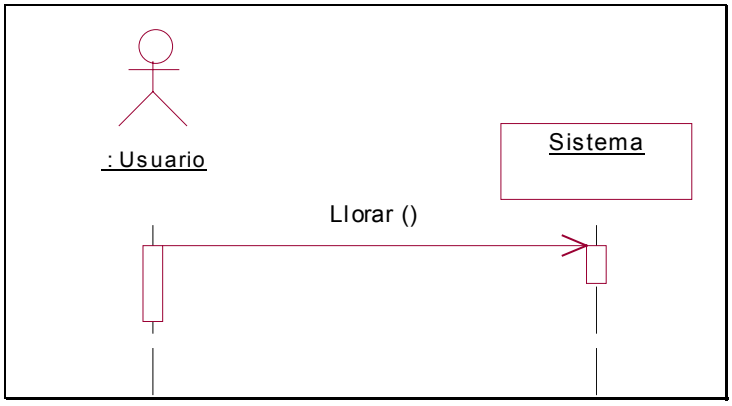

Figura 14: Llorar

1.2.8. Caso de Uso *Ponerse Triste*

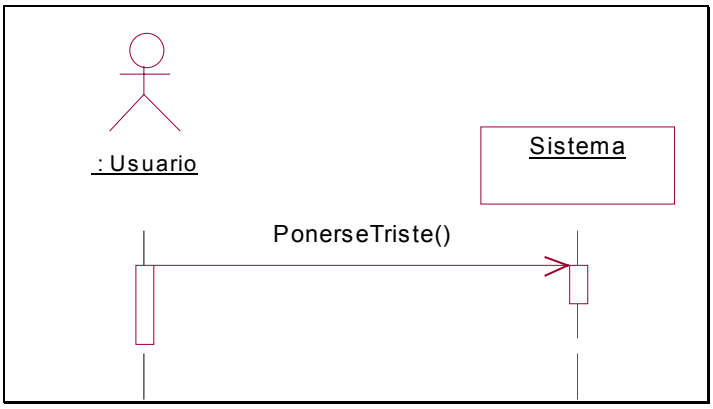

Figura 15: Ponerse Triste

1.2.9. Caso de Uso *Enfadarse*

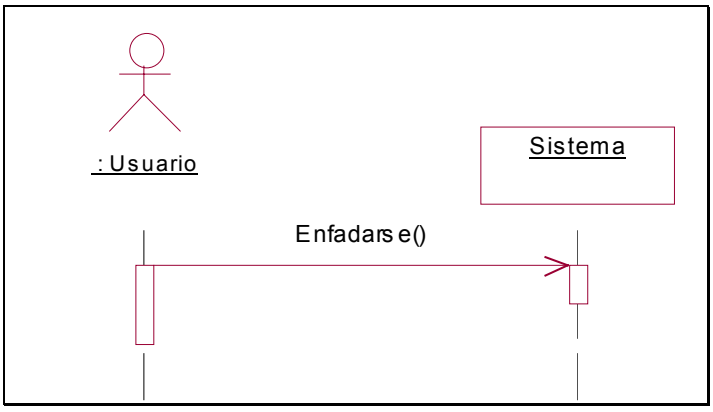

Figura 16: Enfadarse

1.2.10. Caso de Uso *Aburrirse*

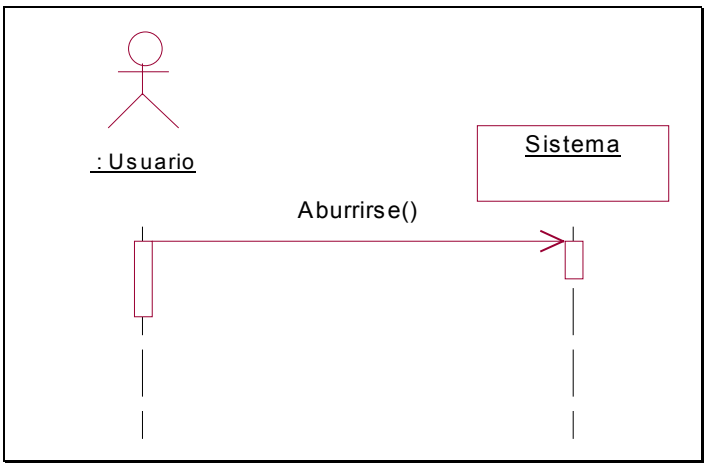

Figura 17: Aburrirse

1.2.11. Caso de Uso *Saltar de Alegría*

Figura 18: Saltar de Alegría

1.2.12. Caso de Uso *Bailar*

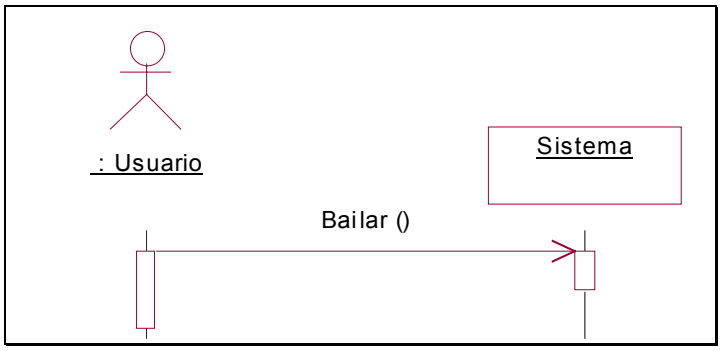

Figura 19: Bailar

1.2.13. Caso de Uso *Patalear*

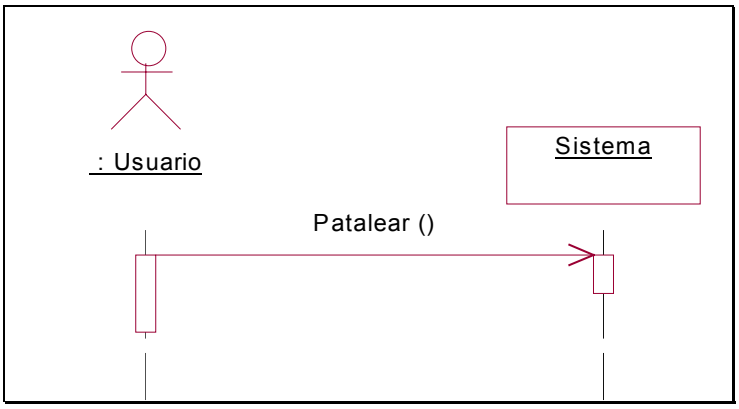

Figura 20: Patalear

1.2.14. Caso de Uso *Comunicar Avatar Visto Moviéndose*

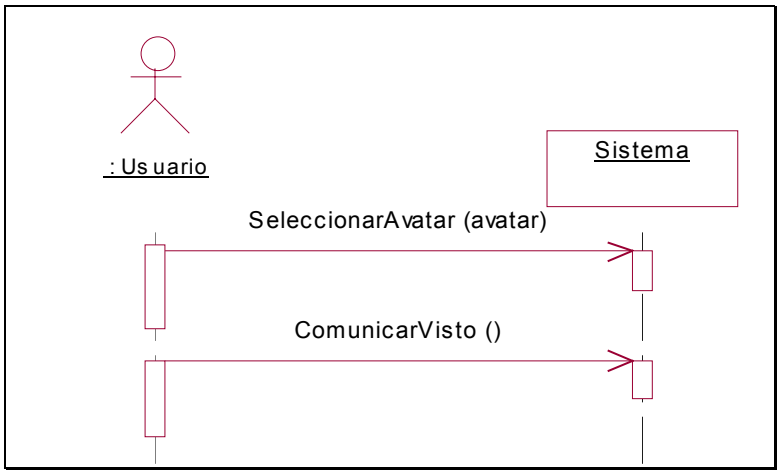

Figura 21: Comunicar Avatar Visto Moviéndose

1.2.15. Caso de Uso *Volver a la Línea de Comienzo*

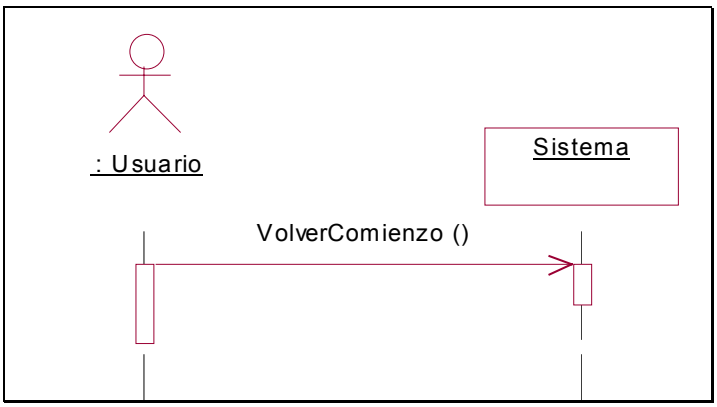

Figura 22: Volver a la Línea de Comienzo

1.2.16. Caso de Uso *Andar*

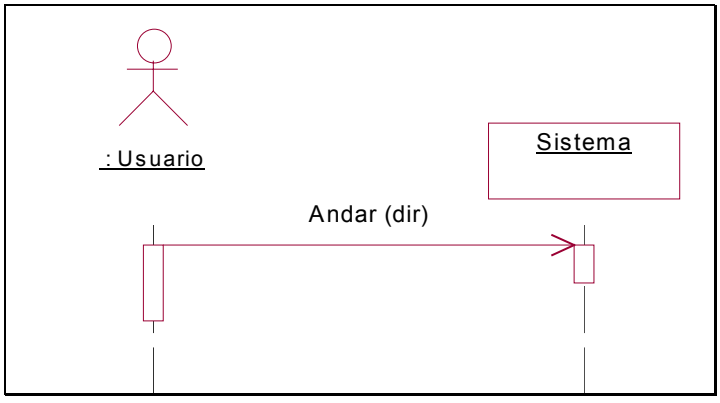

Figura 23: Andar

1.2.17. Caso de Uso *Correr*

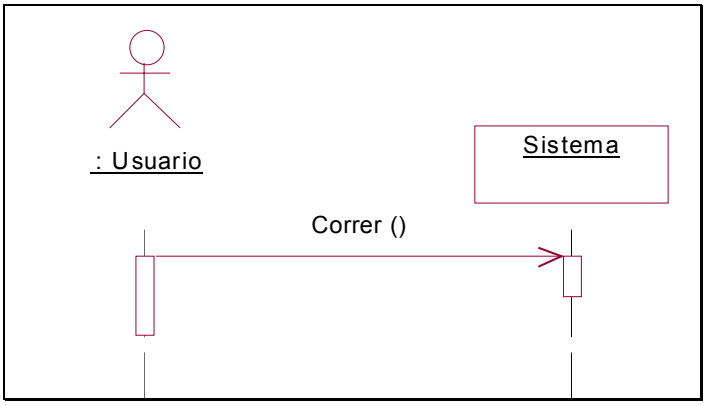

Figura 24: Correr

1.2.18. Caso de Uso *Girar*

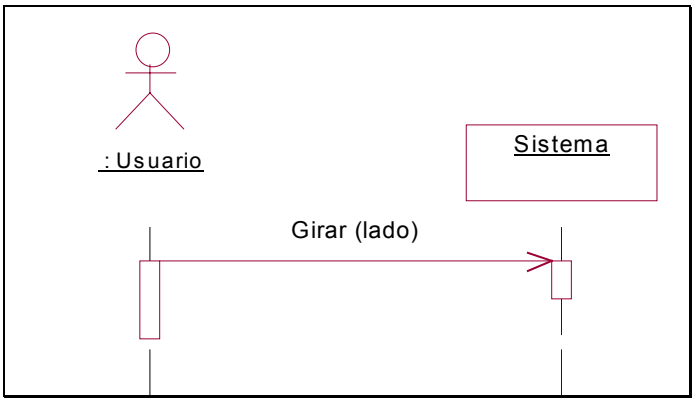

Figura 25: Girar

1.2.19. Caso de Uso *Hablar con otros Usuarios*

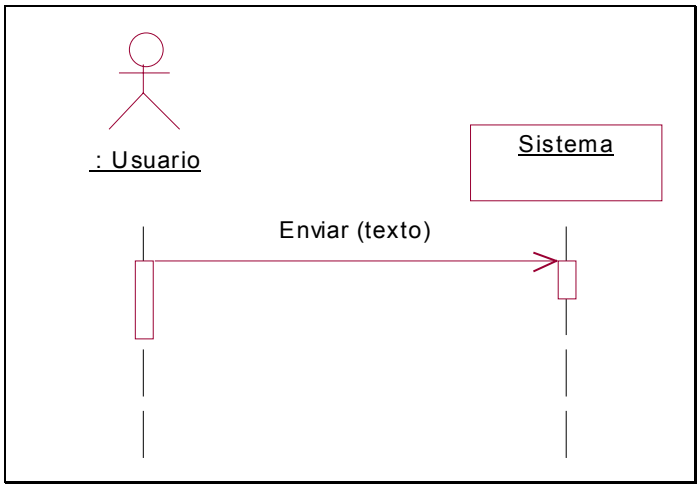

Figura 26: Hablar con otros Usuarios

1.3. MODELO CONCEPTUAL

En este apartado se va a dar una imagen de lo que se ha averiguado hasta ahora del sistema a construir, intentando reflejar también el conocimiento que de este tipo de sistemas se posee.

De esta manera, sabemos, por un lado, que físicamente el EV está compuesto por una serie de objetos visibles o no visibles que van a posibilitar, en su conjunto, que se pueda tener una representación del mismo.

En cuanto a componentes visibles, harán aparición muebles, paredes y puertas, que conformarán el aspecto físico del EV en su vertiente más estática.

En cuanto a proporcionar visibilidad del EV para el usuario, se pueden encontrar dos componentes distintos: por un lado, las luces, que iluminarán las escenas y que permitirán, junto con la decoración, crear distintas atmósferas; por otro lado, los puntos de vista, que serán las ventanas del usuario hacia el EV, y que determinarán la percepción que el usuario tenga del mismo.

Por último, para animar y dar vida al EV, hacen su aparición los avatares, que serán la representación del usuario en el EV. Los avatares estarán dotados de una serie de partes del cuerpo, cuyo número vendrá dado por el grado de detalle con que se quiera representar al avatar dentro del EV. Además, como ya se ha mencionado en etapas anteriores, se deberán contemplar, para el avatar, una serie de rasgos de personalidad y de humor, que determinarán en cierta manera la forma en que éste realizará ciertas acciones. También se sabe que deberá tener una memoria en la que registrará, por ejemplo, una lista de amigos a los que se va conociendo en distintas conexiones al EV.

Por último, también habrá que facilitar una forma de acceder información genérica acerca del desarrollo de la partida y de cada usuario que se ha conectado al EV, en especial del dispositivo que utilizan para moverse por el EV, ya que marcará de manera notoria la percepción que el usuario tenga del EV.

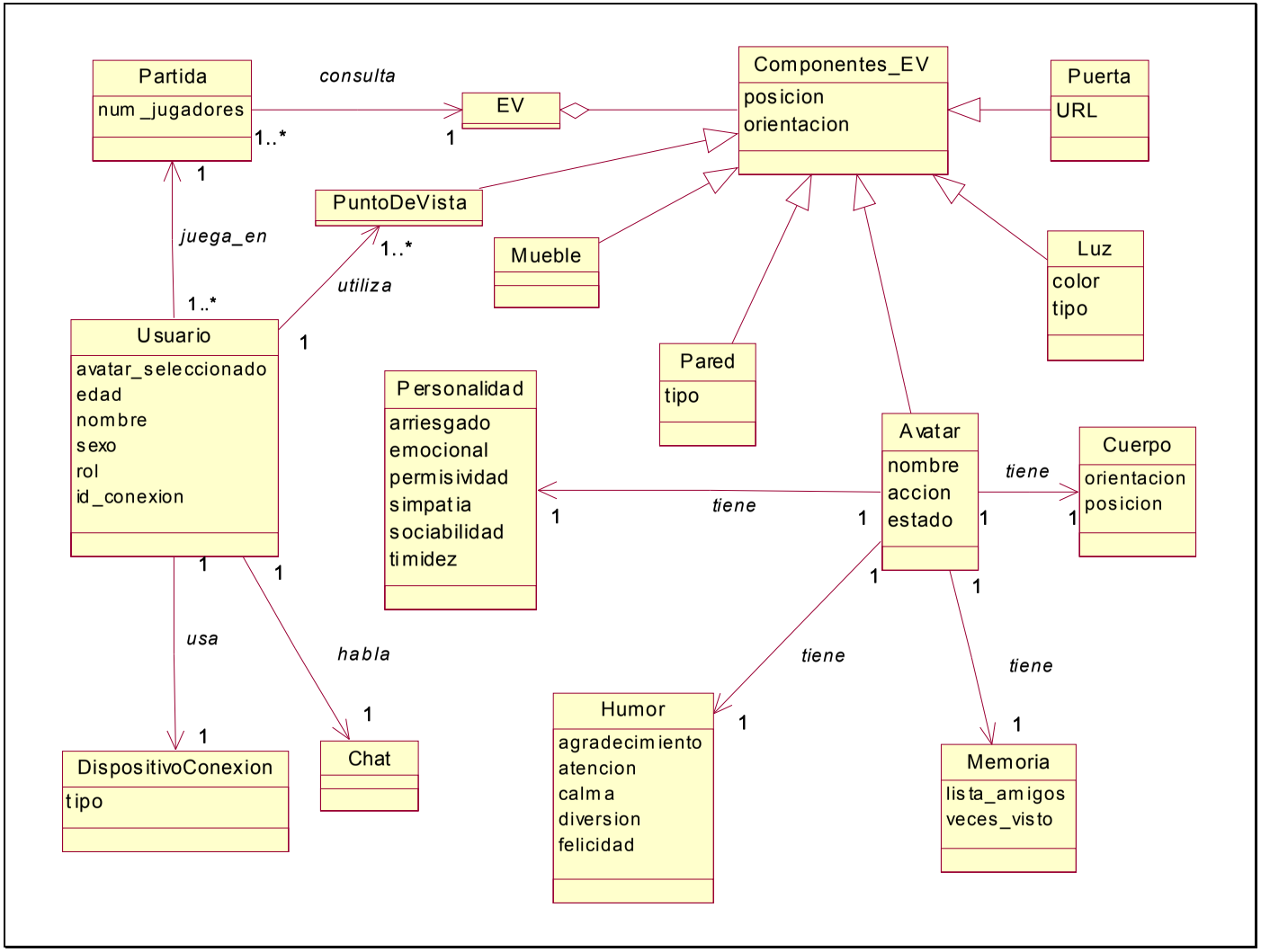

Figura 27: Modelo Conceptual

1.4. CONTRATOS DE OPERACIONES

Nombre	ElegirAvatar (avatar)
Responsabilidades	Marcar uno de los avatares que se le han dado a elegir al usuario para que lo represente cuando haya entrado en el entorno y mostrar la pantalla de selección de puntos de vista.
Tipo	Sistema
Referencias Cruzadas	Caso de uso *Conectarse al Mundo Virtual*.
Excepciones	
Precondiciones	
Postcondiciones	↳ Se ha creado un objeto Avatar, con las partes del Cuerpo correspondientes. ↳ Se ha creado una Memoria, y se ha creado una asociación con Avatar. ↳ Se ha creado un Humor, con valores intermedios. ↳ Se ha creado la relación entre Avatar y Humor. ↳ Se ha creado un *Usuario*.

Nombre	SeleccionarPV (pv)
Responsabilidades	Marcar uno de los puntos de vista que se le ofrecen al usuario para que sea el que utilice cuando entre en el EV y mostrar la pantalla de selección de personalidad.
Tipo	Sistema
Referencias Cruzadas	Caso de uso *Conectarse al Mundo Virtual*.
Excepciones	
Precondiciones	↳ Haber elegido un avatar.
Postcondiciones	↳ Se ha creado un Punto de Vista. ↳ Se ha creado una relación entre el Usuario y el Punto de Vista

Nombre	SeleccionarPersonalidad (pers)
Responsabilidades	Anotar los valores de personalidad que el usuario quiere que tenga su avatar, para asignárselos al entrar en el entorno y mostrar la pantalla de selección de servidor de conexión.
Tipo	Sistema
Referencias Cruzadas	Caso de uso *Conectarse al Mundo Virtual*.
Excepciones	
Precondiciones	↳ Haber seleccionado un punto de vista.
Postcondiciones	↳ Se ha creado una Personalidad con los rasgos seleccionados por el usuario. ↳ Se ha creado una relación entre Avatar y Personalidad.

Nombre	SeleccionarServidor (nombre)
Responsabilidades	Conectarse al servidor de Mundos Virtuales seleccionado para que el usuario pueda interactuar con otros usuarios y mostrar la pantalla de selección de dispositivo de conexión.
Tipo	Sistema
Referencias Cruzadas	Caso de uso *Conectarse al Mundo Virtual*.
Excepciones	Si el servidor de Mundos Virtuales no está disponible, habrá que informar al usuario.
Precondiciones	↳ Haber seleccionado los rasgos de personalidad.
Postcondiciones	

Nombre	SeleccionarDispositivo (disp)
Responsabilidades	Inicializar el dispositivo de conexión utilizado por el usuario y entrar en el EV.
Tipo	Sistema
Referencias Cruzadas	Caso de uso *Conectarse al Mundo Virtual*.
Excepciones	Si no se puede inicializar el dispositivo de conexión, habrá que informar al usuario.
Precondiciones	↳ Haber seleccionado un servidor de conexión.
Postcondiciones	↳ Se ha creado un Dispositivo de Conexión. ↳ Se ha credo la relación del Usuario con el Dispositivo de Conexión. ↳ Se ha creado una Partida. ↳ Se ha creado el EV, con las Luces, Muebles, Paredes y Puertas correspondientes, y se ha creado una relación con la Partida.

Nombre	Contar ()
Responsabilidades	Indicar al avatar que realice la acción de contar.
Tipo	Sistema
Referencias Cruzadas	Caso de uso *Contar*.
Excepciones	
Precondiciones	↳ Que el avatar corresponda al jugador que se la liga. ↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se ha cambiado el valor de la acción del avatar.

Nombre	Saludar ()
Responsabilidades	Indicar al avatar del usuario que salude al avatar seleccionado.
Tipo	Sistema
Referencias Cruzadas	Caso de uso *Saludar*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción. ↳ Que haya un avatar seleccionado.
Postcondiciones	↳ Se ha cambiado el valor de la acción del avatar.

Nombre	SeleccionarAvatar (avatar)
Responsabilidades	Anotar el avatar sobre el que se va a realizar una acción posterior.
Tipo	Sistema
Referencias Cruzadas	Casos de uso *Saludar a otro Avatar, Hacer Reír a otro Avatar, Agredir a otro avatar, Comunicar Avatar Visto Moviéndose*.
Excepciones	
Precondiciones	
Postcondiciones	

Nombre	HacerReir ()
Responsabilidades	Indicar al avatar del usuario que realice la acción de hacer reír al avatar seleccionado.
Tipo	Sistema
Referencias Cruzadas	Caso de uso *Hacer Reír a otro Avatar*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción. ↳ Que haya un avatar seleccionado.
Postcondiciones	↳ Se ha cambiado el valor de la acción del avatar.

Nombre	Agredir ()
Responsabilidades	Indicar al avatar del usuario que realice la acción de agredir sobre el avatar seleccionado.
Tipo	Sistema
Referencias Cruzadas	Caso de uso *Agredir a otro Avatar*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción. ↳ Que haya un avatar seleccionado.
Postcondiciones	↳ Se ha cambiado el valor de la acción del avatar.

Nombre	Reir ()
Responsabilidades	Indicar al avatar del usuario que realice la acción de reír.
Tipo	Sistema
Referencias Cruzadas	Caso de uso *Reír*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se ha cambiado el valor de la acción del avatar. ↳ Se han cambiado los valores del humor del avatar.

Nombre	Llorar ()
Responsabilidades	Indicar al avatar del usuario que realice la acción de llorar.
Tipo	Sistema
Referencias Cruzadas	Caso de uso *Llorar*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se ha cambiado el valor de la acción del avatar. ↳ Se han cambiado los valores del humor del avatar.

Nombre	PonerseTriste ()
Responsabilidades	Indicar al avatar del usuario que realice la acción de ponerse triste.
Tipo	Sistema
Referencias Cruzadas	Caso de uso *Ponerse Triste*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se ha cambiado el valor de la acción del avatar. ↳ Se han cambiado los valores del humor del avatar.

Nombre	Enfadarse ()
Responsabilidades	Indicar al avatar del usuario que realice la acción de enfadarse.
Tipo	Sistema
Referencias Cruzadas	Caso de uso *Enfadarse*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se ha cambiado el valor de la acción del avatar. ↳ Se han cambiado los valores del humor del avatar.

Nombre	Aburrirse ()
Responsabilidades	Indicar al avatar del usuario que realice la acción de aburrirse.
Tipo	Sistema
Referencias Cruzadas	Caso de uso *Aburrirse*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se ha cambiado el valor de la acción del avatar. ↳ Se han cambiado los valores del humor del avatar.

Nombre	SaltarDeAlegria ()
Responsabilidades	Indicar al avatar del usuario que realice la acción de saltar de alegría.
Tipo	Sistema
Referencias Cruzadas	Caso de uso *Saltar de Alegría*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se ha cambiado el valor de la acción del avatar. ↳ Se han cambiado los valores del humor del avatar.

Nombre	Bailar ()
Responsabilidades	Indicar al avatar del usuario que realice la acción de bailar.
Tipo	Sistema
Referencias Cruzadas	Caso de uso *Bailar*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se ha cambiado el valor de la acción del avatar. ↳ Se han cambiado los valores del humor del avatar.

Nombre	Patalear ()
Responsabilidades	Indicar al avatar del usuario que realice la acción de patalear.
Tipo	Sistema
Referencias Cruzadas	Caso de uso *Patalear*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se ha cambiado el valor de la acción del avatar. ↳ Se han cambiado los valores del humor del avatar.

Nombre	ComunicarVisto ()
Responsabilidades	Indicar al avatar seleccionado que se le ha visto moviéndose.
Tipo	Sistema
Referencias Cruzadas	Caso de uso *Avatar Visto Moviéndose*.
Excepciones	
Precondiciones	↳ Que haya un avatar seleccionado.
Postcondiciones	

Nombre	VolverComienzo ()
Responsabilidades	Indicar al avatar del usuario que vuelva a la línea de comienzo.
Tipo	Sistema
Referencias Cruzadas	Caso de uso *Volver a la línea de comienzo*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	

Nombre	Andar (dir)
Responsabilidades	Indicar al avatar del usuario que realice la acción de andar en la dirección seleccionada.
Tipo	Sistema
Referencias Cruzadas	Caso de uso *Andar*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción, salvo correr o girar.
Postcondiciones	↳ Se ha cambiado el valor de la acción del avatar.

Nombre	Correr ()
Responsabilidades	Indicar al avatar del usuario que realice la acción de correr.
Tipo	Sistema
Referencias Cruzadas	Caso de uso *Correr*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción, salvo andar o girar.
Postcondiciones	↳ Se ha cambiado el valor de la acción del avatar.

Nombre	Girar (lado)
Responsabilidades	Indicar al avatar del usuario que realice la acción de girar hacia el lado seleccionado.
Tipo	Sistema
Referencias Cruzadas	Caso de uso *Girar*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción, salvo correr o andar.
Postcondiciones	↳ Se ha cambiado el valor de la acción del avatar.

Nombre	Enviar (texto)
Responsabilidades	Enviar al resto de los usuarios el texto que se ha escrito.
Tipo	Sistema
Referencias Cruzadas	Caso de uso *Hablar con otros Usuarios*.
Excepciones	
Precondiciones	
Postcondiciones	

1.5. GLOSARIO

Ahora es el momento, una vez que se tiene construido el modelo conceptual, de ampliar el glosario que se ha visto en la fase de Planificación y Especificación de Requisitos.

Término	Categoría	Descripción
Aburrirse ()	*Operación*	Indica al avatar del usuario que realice la acción de aburrirse.
Agredir ()	*Operación*	Indica al avatar del usuario que realice la acción de agredir sobre el avatar seleccionado.
Andar (dir)	*Operación*	Indica al avatar del usuario que realice la acción de andar en la dirección seleccionada.
Avatar	*Concepto*	Representa, de manera abstracta, a los avatares que puede utilizar un usuario para moverse por el EV.
Avatar.accion	*Atributo*	Acción que está realizando un avatar en un momento dado.
Avatar.estado	*Atributo*	Estado en que se encuentra la realización de una acción.
Avatar.nombre	*Atributo*	Nombre del avatar.
Bailar ()	*Operación*	Indica al avatar del usuario que realice la acción de bailar.
Chat	*Concepto*	Herramienta utilizada por los usuarios para comunicarse entre ellos.
Componentes_EV	*Concepto*	Elementos de los que estará constituido un EV.
Componentes_EV.orientacion	*Atributo*	Orientación del componente dentro del EV.
Componentes_EV.posicion	*Atributo*	Posición del componente dentro del EV.
ComunicarVisto ()	*Operación*	Indica al avatar seleccionado que se le ha visto moviéndose.
Contar ()	*Operación*	Indica al avatar que realice la acción de contar.
Correr ()	*Operación*	Indica al avatar del usuario que realice la acción de correr.
Cuerpo	*Concepto*	Representación física del cuerpo del avatar.
Cuerpo.orientacion	*Atributo*	Orientación del cuerpo del avatar.
Cuerpo.posicion	*Atributo*	Posición del cuerpo del avatar.
DispositivoConexion	*Concepto*	Dispositivo que utiliza el usuario para moverse por el EV (cámara o teclado).
DispositivoConexion.tipo	*Atributo*	Tipo de dispositivo de conexión que utiliza el usuario.
ElegirAvatar (avatar)	*Operación*	Marca uno de los avatares que se le han dado a elegir al usuario para que lo represente cuando haya entrado en el entorno y mostrar la pantalla de selección de puntos de vista.

Enfadarse ()	*Operación*	Indica al avatar del usuario que realice la acción de enfadarse.
Enviar (texto)	*Operación*	Enviar al resto de los usuarios el texto que se ha escrito.
EV	*Concepto*	Representa el Entorno Virtual al que se conectan los usuarios para jugar al escondite inglés.
Girar (lado)	*Operación*	Indica al avatar del usuario que realice la acción de girar hacia el lado seleccionado.
HacerReir ()	*Operación*	Indica al avatar del usuario que realice la acción de hacer reír al avatar seleccionado.
Humor	*Concepto*	Recoge los diferentes aspectos que forman el humor en que se encuentra un avatar. Sirve para modificar la conducta del mismo.
Humor.agradecimiento	*Atributo*	Grado de agradecimiento del avatar. Varía entre agradecido y enfadado.
Humor.atencion	*Atributo*	Grado de atención del avatar. Varía entre atento y poco atento.
Humor.calma	*Atributo*	Grado de nerviosismo del avatar. Varía entre calmado y excitado.
Humor.diversion	*Atributo*	Grado de diversión de un avatar. Varía entre divertido y aburrido.
Humor.felicidad	*Atributo*	Grado de felicidad del avatar. Varía entre feliz y triste.
Llorar ()	*Operación*	Indica al avatar del usuario que realice la acción de llorar.
Luz	*Concepto*	Fuente de luz necesaria para iluminar el EV.
Luz.color	*Atributo*	Color de la luz.
Luz.tipo	*Atributo*	Tipo de luz: ambiental, dirigida, etc.
Memoria	*Concepto*	Permite que un avatar recuerde cosas sucedidas en otras conexiones al EV.
Memoria.lista_amigos	*Atributo*	Amigos que ha hecho un avatar a lo largo de distintas conexiones al EV.
Memoria.veces_visto	*Atributo*	Número de veces que el avatar ha sido visto moviéndose.
Mueble	*Concepto*	Cualquier objeto inanimado que pueda aparecer en una habitación del EV.
Pared	*Concepto*	Las paredes delimitan las estancias del EV.
Pared.tipo	*Atributo*	Indica si la pared cumple alguna función especial, como la pared donde cuenta el jugador que se la liga en el escondite inglés.
Partida	*Concepto*	Representa al juego al que se estará jugando, en general, dentro del EV.
Partida.num_jugadores	*Atributo*	Número de jugadores que se encuentran jugando una partida dentro del EV.
Patalear ()	*Operación*	Indica al avatar del usuario que realice la acción de patalear.

Personalidad	*Concepto*	Recoge los aspectos que forman la personalidad del avatar.
Personalidad.arriesgado	*Atributo*	Grado de riesgo que es capaz de asumir un avatar en sus acciones.
Personalidad.emocional	*Atributo*	Grado de emocionalidad de un avatar. Varía de muy emocional a nada emocional.
Personalidad.permisividad	*Atributo*	Indica el grado de permisividad de un avatar. Varía entre muy permisivo y nada permisivo.
Personalidad.simpatia	*Atributo*	Grado de simpatía de un avatar. Varía entre muy simpático y nada simpático.
Personalidad.sociabilidad	*Atributo*	Grado de sociabilidad de un avatar. Varía entre muy sociable y nada sociable.
Personalidad.timidez	*Atributo*	Grado de timidez de un avatar. Varía entre muy tímido y nada tímido.
PonerseTriste ()	*Operación*	Indica al avatar del usuario que realice la acción de ponerse triste.
Puerta	*Concepto*	Es un punto de entrada y de salida del EV en el que nos encontremos.
Puerta.URL	*Atributo*	Lugar al que se llega al atravesar una puerta.
Reir ()	*Operación*	Indica al avatar del usuario que realice la acción de reír.
SaltarDeAlegria ()	*Operación*	Indica al avatar del usuario que realice la acción de saltar de alegría.
Saludar ()	*Operación*	Indica al avatar del usuario que salude al avatar seleccionado.
SeleccionarAvatar (avatar)	*Operación*	Anota el avatar sobre el que se va a realizar una acción posterior.
SeleccionarDispositivo (disp)	*Operación*	Inicializa el dispositivo de conexión utilizado por el usuario y entrar en el EV.
SeleccionarPersonalidad (pers)	*Operación*	Anota los valores de personalidad que el usuario quiere que tenga su avatar, para asignárselos al entrar en el entorno y mostrar la pantalla de selección de servidor de conexión.
SeleccionarPV (pv)	*Operación*	Marca uno de los puntos de vista que se le ofrecen al usuario para que sea el que utilice cuando entre en el EV y mostrar la pantalla de selección de personalidad.
SeleccionarServidor (nombre)	*Operación*	Se conecta al servidor de Mundos Virtuales seleccionado para que el usuario pueda interactuar con otros usuarios y mostrar la pantalla de selección de dispositivo de conexión.
Usuario	*Concepto*	Representa a la persona que, a través de un avatar, se introduce en el EV para interactuar con otros usuarios.
Usuario.avatar_seleccionado	*Atributo*	Avatar elegido por el usuario.
Usuario.edad	*Atributo*	Edad del usuario.
Usuario.id_conexion	*Atributo*	Identificador del usuario para poder distinguirlo de los demás.

Usuario.nombre	*Atributo*	Nombre del usuario.
Usuario.rol	*Atributo*	Rol que desempeña el usuario en la partida (se la liga o juega).
Usuario.sexo	*Atributo*	Sexo del usuario.
VolverComienzo ()	*Operación*	Indica al avatar del usuario que vuelva a la línea de comienzo.

2. DISEÑO

En esta fase del método de Larman [Larman99] hay que contemplar, además de todo lo que está descrito, el diseño 3D del EV. Sin embargo, este aspecto del diseño no se contempla dentro de lo descrito por Larman, y tampoco es fácil encuadrarlo dentro de ninguna de las fases existentes, como podría ser el diseño de la interfaz, ya que posee la suficiente entidad como para que haya aspectos que no se puedan considerar en las fases que describe Larman.

Por esta razón, se ha decidido establecerla como una fase separada dentro de la fase de diseño, situada además antes que el resto, ya que la manera en que se haga el diseño 3D del escenario y los avatares determinará parte del diseño del comportamiento de las distintas acciones que se puedan realizar, en concreto todas las que requieran un movimiento por parte del avatar.

Puesto que, como se ha dicho, este aspecto del diseño no ha sido considerado por Larman, el diseño 3D del EV se ha realizado siguiendo las indicaciones expuestas en [Sánchez99], según se presenta a continuación.

2.1. DISEÑO 3D DEL EV

El modelado de un EV conlleva la toma de muchas decisiones que afectarán tanto al aspecto del EV como a la velocidad de renderización y al tamaño de éste. Los diseñadores gráficos no tienen por qué estar familiarizados con la problemática que surge a la hora de construir EV multiusuario en red; por este motivo los diseñadores han de recibir una serie de directrices a la hora de construir el EV.

El primer paso consiste en la descripción del tamaño y elementos básicos que componen el espacio "real" en el que el usuario podrá moverse.

2.1.1. Tamaño del Escenario y Disposición de sus Elementos

En la Figura 1, veíamos el escenario de forma general; ahora, en la Figura 28, vamos a describirlo indicando las dimensiones y la posición de cada elemento, teniendo en cuenta el tipo de cámara que se ha seleccionado. Las distancias vienen expresadas en centímetros.

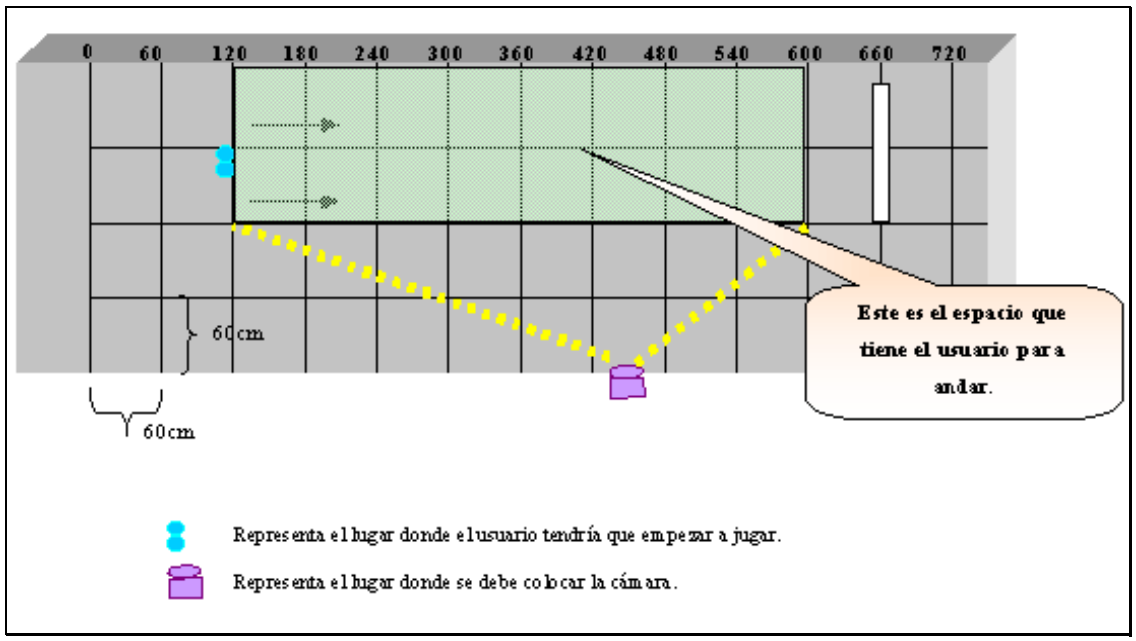

Figura 28: Medidas del Escenario Real

2.1.2. Mapa de Vistas

Como segundo paso a seguir, se debe crear un mapa que constará de tres tipos de vistas como mínimo, de manera que se pueda especificar claramente qué objetos habrá en el EV y cómo estarán colocados [Sánchez99]. Las vistas correspondientes al juego del Escondite Inglés son las que se presentan a continuación:

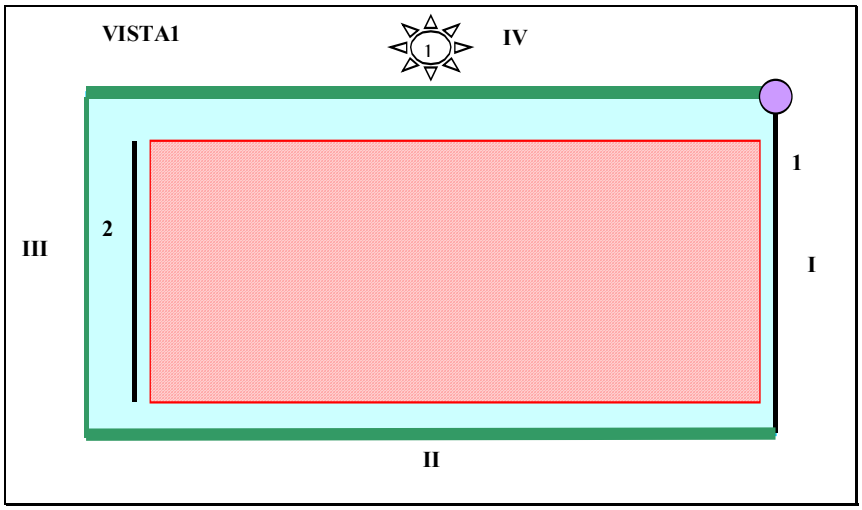

Figura 29: VISTA1 del sistema

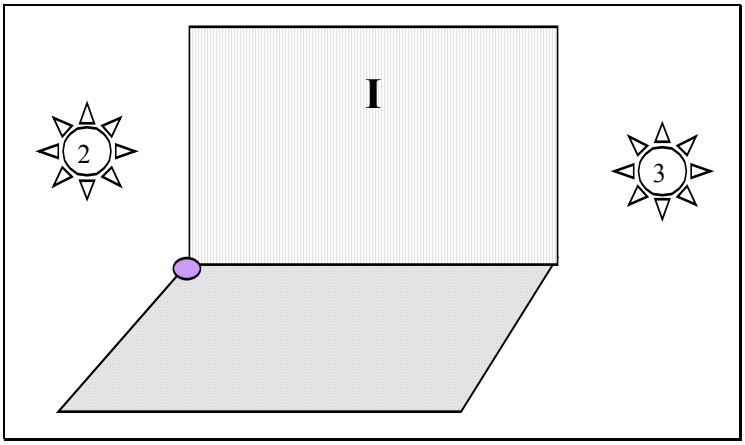

Figura 30: VISTA2 del sistema

Existe un tercer tipo de vista para representar objetos que cuelgan del techo, pero, puesto que no van a existir en el escenario objetos de este tipo, no es necesario incluirla.

2.1.3. Formulario de Modelado

A continuación, se debe rellenar un cuestionario básico con las características que debe tener el EV y que deberán ser respetadas por el diseñador gráfico [Sánchez99]. En el propio formulario se podrá añadir todo tipo de comentarios que puedan ayudar al diseñador gráfico en su labor. Además, se debe construir un pequeño mapa con una simbología conocida por el diseñador gráfico para la determinación de posibles espacios vacíos, ubicación concreta de ciertos objetos, etc.

No es obligatorio rellenar ni todos los campos del formulario ni todos los atributos de cada campo, pero cuanta más información se le proporcione al diseñador gráfico menos probabilidad hay de incurrir en errores debidos a la falta de entendimiento.

A continuación aparece el formulario para el presente desarrollo:

<table>
<tr><td colspan="6">FORMULARIO DE MODELADO DEL EV: Escondite Inglés</td></tr>
<tr><td colspan="2">Elementos obligatorios:</td><td colspan="4">Nombre: Pared de fondo
Descripción: Lugar donde se situará el que se la liga para contar.
Número: 1
Posición: Ver Vista1.
Nombre: Línea en el suelo
Descripción: Lugar del que parten los jugadores. Debe ser una marca en el suelo.
Número: 2
Posición: Ver Vista1.</td></tr>
<tr><td colspan="6">Tipo de ornamentación: Sencilla</td></tr>
<tr><td colspan="6">Tipo de decorado: Lugar donde pueda haber espacio para desarrollar un juego en el que se corre.</td></tr>
<tr><td colspan="2">El EV tendrá techo:</td><td colspan="4">Sí:
No: no necesariamente, se puede ver el cielo al mirar hacia arriba.</td></tr>
<tr><td colspan="2">El EV tendrá suelo:</td><td colspan="4">Sí: para que se perciba más claramente la distancia de cada avatar a la pared de meta.
No:</td></tr>
<tr><td colspan="2">Tamaño del entorno virtual:</td><td colspan="2">Condicionado
X</td><td colspan="2">No condicionado</td></tr>
<tr><td colspan="3">El EV podrá tener columnas:</td><td colspan="2">Sí X
Pero sólo en las paredes laterales</td><td>No</td></tr>
<tr><td colspan="3">El entorno virtual podrá tener texturas:</td><td colspan="2">Sí X</td><td>No</td></tr>
<tr><td colspan="3">Posición de los ejes en la herramienta de desarrollo</td><td>X arriba</td><td>Y arriba
X</td><td>Z arriba</td></tr>
<tr><td colspan="6">Formato de exportación: ficheros propios de Alias Power Animator, .obj</td></tr>
<tr><td colspan="6">Tipo de exportación:</td></tr>
<tr><td rowspan="2">Polígonos X</td><td>Triángulos X</td><td rowspan="3">Número de polígonos:</td><td>< 500</td></tr>
<tr><td rowspan="2">Cuadrados</td><td>> 500 y < 1000 X</td></tr>
<tr><td rowspan="2">Curvas</td><td rowspan="2">Tipo de curva</td><td>> 1000</td></tr>
<tr><td>no existe restricción</td></tr>
</table>

2.2. DISEÑO DE LA ESTRUCTURA FÍSICA DE LOS AVATARES

Antes de pasar al diseño del software que nos ocupa, tenemos que pararnos a diseñar cómo van a ser los avatares que van a habitar el EV. Se trata de una cuestión nada trivial; de hecho, existen grupos de investigación cuyo objetivo es el de estandarizar la estructura de los avatares.

En un desarrollo de este tipo se gasta mucho tiempo en la creación de los avatares; este es el motivo por el que sería ideal que los avatares tuviesen una estructura tal que cualquiera que los necesitase pudiera manejarlos dentro de su EV, con lo que evitaríamos repetir mucho trabajo inútil. La idea base sería la reutilización, lo cual concuerda bastante con la utilización de una metodología orientada a objetos como es la utilizada en el presente desarrollo.

De todo lo especificado anteriormente, se deduce que el tipo de avatar idóneo es el avatar de cuerpo completo, capaz de llevar a cabo todo lo descrito durante el análisis, bien de forma automática o bajo demanda del usuario.

Por este motivo, consideramos que en el presente desarrollo se debe hacer una reflexión sobre qué estándares hay y si podemos usarlos.

2.2.1. Los Movimientos Humanos

En cuanto al diseño de los movimientos humanos 3D, podemos hacer una distinción que viene determinada por la forma en que éstos son generados y el tiempo que se ha asignado para su elaboración. Los movimientos pueden ser creados off-line, por un diseñador, con el fin de conseguir un alto grado de credibilidad, o pueden ser generados en tiempo real como resultado de una interacción. Este último será el mecanismo que se utilice para realizar los movimientos de los avatares.

2.2.2. Avatares de Cuerpo Completo

A medida que los entornos virtuales en Internet crecen, existe una necesidad cada vez mayor de representar avatares que pueblen estos EV.

Si el objetivo de estos entornos es que la gente se conecte para visitar lugares nuevos, fomentar las relaciones sociales, etc., parece necesario que, además de poder adquirir cualquier tipo de apariencia, también podamos ser representados por avatares

similares a los humanos, al menos si queremos experimentar con interfaces de captura de movimiento.

Un gran número de personas trabajan en la definición de estándares, como Living World y VRML Humanoid Animation Working Group, pero realmente no hay nada firmemente establecido, excepto la especificación de VRML97, que hace referencia específica a mundos virtuales multiusuario.

Existe un gran número de proyectos e investigaciones que se basan directamente en la creación de "humanos virtuales" o avatares. El grupo "Virtual Humans Architecture Group (V-HAG)", se creó para unificar algunas de las investigaciones clave llevadas a cabo en este campo. El objetivo es identificar cuál es la necesidad común, a través de todas las tentativas de estándares que hay, y una vez identificada cuál es la necesidad, unificar esfuerzos con el fin de coordinar un estándar que sirva para todas las áreas en las que los avatares, como representación de los humanos en los EV, son útiles.

2.2.3. Estándares existentes

A continuación presentamos las dos iniciativas más ampliamente difundidas, relacionadas con "Humanos virtuales":

- **MPEG-4 Synthetic/Natural Hybrid Coding (SNHC)**: Los ficheros MPEG se basan en secuencias de bits que deben ser decodificadas. Estas secuencias de bits contienen la definición del cuerpo de un avatar, y parámetros de animación del mismo. MPEG basa la topología del cuerpo del avatar en el esqueleto humano, estableciendo un grano muy fino en la jerarquía. Proponen seis grados de libertad (DOF, *Degree Of Freedom*) para animar el cuerpo del avatar, y 66 DOF para movilidad de pequeñas articulaciones.
- **Universal Avatars Proposal,** concretamente **H-ANIM (Humanoid Animation Group):** H-ANIM es una especificación para el estándar "VRML Humanoid", y se basa en el estándar VRML 2.0. Al igual que MPEG, HANIM se basa en el esqueleto humano, definiendo un anidamiento en la jerarquía bastante detallado, estableciendo tres niveles de articulación. Los ficheros del estándar VRML Humanoid contienen un conjunto de nodos que constituyen la jerarquía.

2.2.4. Avatares en el presente desarrollo

Existen algunos EV en los que es conveniente definir avatares de cuerpo completo, dada la naturaleza del EV. Este es el caso del EV que nos ocupa, ya que los avatares han de ser capaces de saludarse, andar, correr, etc., respondiendo siempre al dispositivo de interfaz seleccionado.

Los dos estándares mencionados antes son muy completos, pero al mismo tiempo demasiado complejos para nuestro propósito. Este es el motivo por el que decidimos utilizar una definición de los avatares mucho más sencilla, pero también basada en el esqueleto humano, que se puede encontrar en [Sánchez99].

2.2.5. Modelado de los cuerpos virtuales

El esqueleto virtual que hemos descrito se basa en la estructura anatómica del esqueleto real, y viene representado por una arquitectura tridimensional articulada.

A continuación vamos a describir el esqueleto virtual; en la Figura 31 podemos ver la jerarquía general. Se puede observar que el tronco tiene dependencia con todo el resto de elementos del esqueleto.

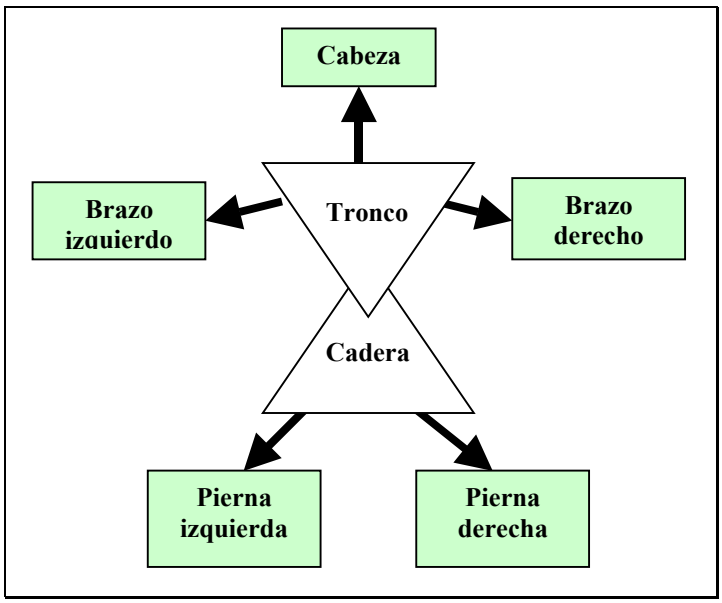

Figura 31: Jerarquía general

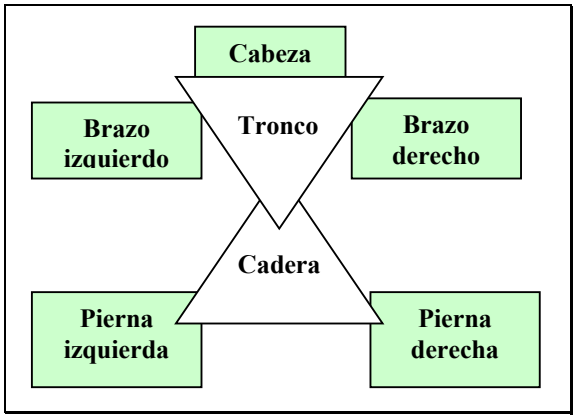

Figura 32: Esqueleto con las articulaciones

Ahora presentamos cada elemento de los mencionados antes, con sus componentes. En la Figura 33 podemos ver que el cuello es un elemento independiente de la cabeza; esto se debe a que existen algunos movimientos, de rotación por ejemplo, que implican a la cabeza pero no al cuello.

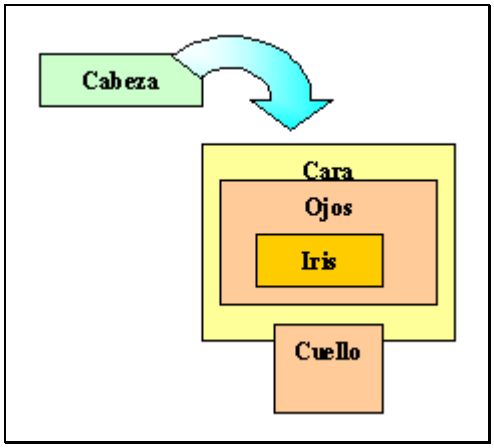

Figura 33: Jerarquía de la cabeza

Como podemos ver en la Figura 34, los hombros aparecen separados del resto del brazo, ya que estos, por sí solos, son un elemento bastante expresivo, por ejemplo para indicar tristeza.

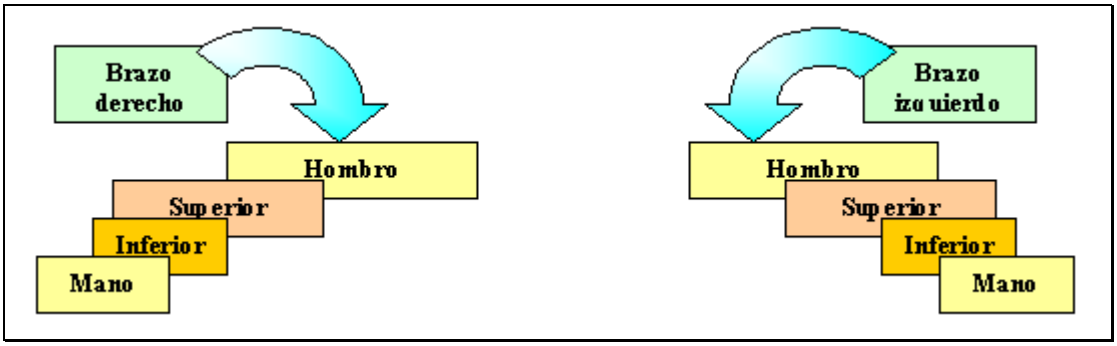

Figura 34: Jerarquía de los brazos

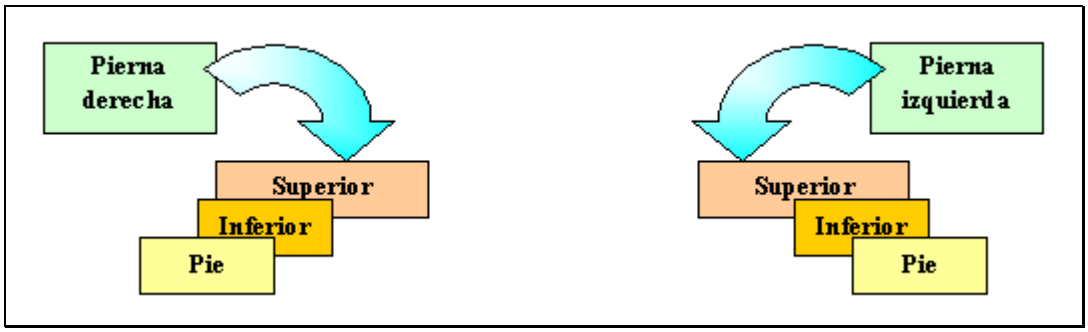

Figura 35: Jerarquía de las piernas

Del mismo modo que se rellenó un formulario para que el diseñador gráfico tenga una idea un poco más clara del entorno que se desea construir, es necesario rellenar uno similar para la construcción de los avatares que van a poblar dicho entorno. En el caso en que puedan existir distintos tipos de avatares se debe rellenar un formulario distinto por tipo de avatar [Sánchez99].

El formulario en cuestión tiene algunos campos iguales a los del formulario de Modelado del EV. Otros son específicos de los avatares y hacen referencia a las partes móviles, puntos de inflexión, etc. Todo esto viene determinado por los diferentes movimientos que tenga que hacer el avatar y que se habrán identificado en la especificación de requisitos.

El motivo de especificar claramente los puntos del cuerpo en los que debe existir articulación es el de indicarle al diseñador gráfico cuáles son los objetos que han de exportarse por separado, de modo que, tras la importación del avatar desde la herramienta de desarrollo, queden separados los objetos han de rotar sobre otros, etc.

A continuación aparece el formulario para los avatares del Escondite Inglés.

<table>
<tr><td colspan="6">FORMULARIO DE MODELADO DE LOS AVATARES</td></tr>
<tr><td rowspan="6">Indique si el avatar dispondrá o no de los siguientes componentes</td><td>Cabeza SI</td><td colspan="4" rowspan="6">Si existe algún otro componente del avatar que deba ser especificado indíquelo en este apartado..</td></tr>
<tr><td>Tronco SI</td></tr>
<tr><td>Brazo izquierdo SI</td></tr>
<tr><td>Brazo derecho SI</td></tr>
<tr><td>Pierna izquierda SI</td></tr>
<tr><td>Pierna derecha SI</td></tr>
<tr><td rowspan="6">Indique si el avatar requiere articulación en los siguiente puntos</td><td>Cuello SI</td><td colspan="4" rowspan="6">Si existe algún otro punto donde se requiera articulación indíquese en este apartado.</td></tr>
<tr><td>Codo SI</td></tr>
<tr><td>Muñecas SI</td></tr>
<tr><td>Cintura SI</td></tr>
<tr><td>Hombro SI</td></tr>
<tr><td>Rodilla SI</td></tr>
<tr><td>Tobillo SI</td></tr>
<tr><td colspan="6">Restricciones en cuanto al tamaño del avatar en relación con el EV</td></tr>
<tr><td colspan="2">El avatar podrá tener texturas:</td><td colspan="2">Sí X</td><td colspan="2">No</td></tr>
<tr><td>Posición de los ejes en la herramienta de desarrollo</td><td>X arriba</td><td>Y arriba
X</td><td>Z arriba</td><td colspan="2"></td></tr>
<tr><td colspan="6">Formato de exportación: ficheros propios de Alias Power Animator, .obj</td></tr>
<tr><td colspan="6">Tipo de exportación:</td></tr>
<tr><td rowspan="4">Polígonos X</td><td>Triángulos X</td><td rowspan="4">Cuadrados</td><td rowspan="4">Número de polígonos:</td><td>< 500 X</td></tr>
<tr><td>> 500 y < 1000</td></tr>
<tr><td>> 1000</td></tr>
<tr><td>no existe restricción</td></tr>
<tr><td>Curvas</td><td>Tipo de curva</td><td colspan="4"></td></tr>
</table>

En la Figura 36, podemos ver la estructura de uno de nuestros avatares, incluyendo los diferentes elementos mencionados antes.

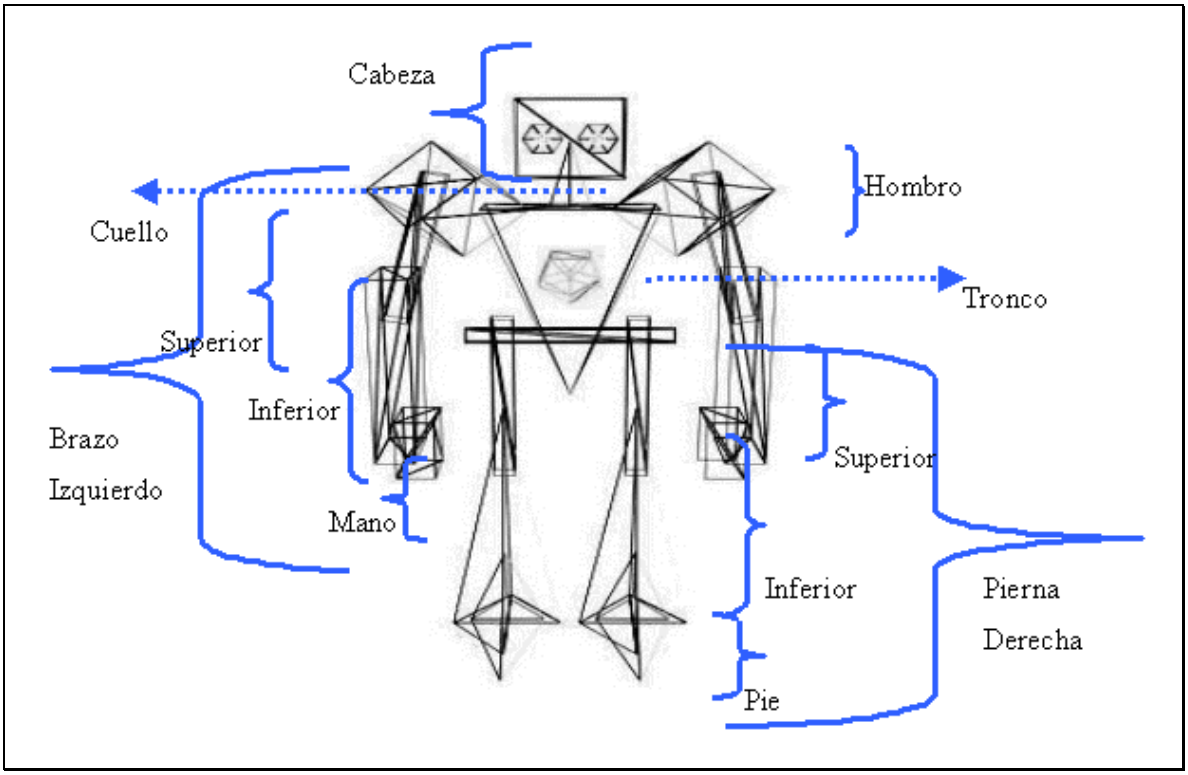

Figura 36: Avatar

2.3. DEFINICIÓN DE LA INTERFAZ DE USUARIO

La interfaz de usuario depende básicamente del dispositivo de captura de movimiento que se seleccione.

En el caso de seleccionar el teclado como dispositivo, se observará el EV a través de la pantalla del ordenador, y se utilizarán las siguientes teclas para mover el avatar:

W: para ordenar que el avatar ande.

S: para ordenar al avatar que corra.

X: para ordenar al avatar que se pare.

G: para caminar hacia atrás.

Además, habrá una serie de botones para que, en el caso en que le toque ligársela, pueda indicarle a su avatar que se dé la vuelta. Una vez que termine de contar, es decir, termine de decir la frase propia del juego, *"Un, dos, tres, al escondite ingles sin mover las manos ni los pies"*, el avatar se dará la vuelta y quedará mirando a los jugadores.

Si se ha seleccionado la cámara como dispositivo de captura de movimiento, es necesario tener los componentes propuestos en la Figura 1. El usuario sólo tendrá que moverse por delante de la cámara para jugar. Si a pesar de haber seleccionado la cámara le toca ligársela, no tendrá más remedio que utilizar los botones que se habrán introducido en la interfaz para tal fin.

Para ambos tipos de interfaz, la pantalla del ordenador queda reducida a una pequeña barra de botones en la que aparece el botón de mirar a la pared para contar, en caso de que el jugador se la ligue, y una serie de botones para realizar acciones como bailar, saltar de alegría, etc. En el resto de la pantalla se visualiza el contenido del EV.

Desde el momento en que un usuario se conecta al EV y hasta que aparece representado en éste, el flujo de acontecimientos que se suceden está preestablecido y es el siguiente, según se definió en el caso de uso *Conectarse al Mundo Virtual*:

1. El usuario arranca el EV.
2. El sistema pregunta al usuario qué apariencia desea tener en el EV.
3. El sistema asigna avatar al usuario.
4. El sistema pregunta al usuario qué punto de vista desea utilizar.
5. El sistema asigna un punto de vista al usuario.
6. El sistema solicita al usuario que dé valores a ciertos rasgos de su personalidad.
7. El usuario contesta, y dichos valores son almacenados por el sistema en los rasgos correspondientes.
8. El sistema pregunta al usuario a qué servidor desea conectarse.
9. El usuario selecciona el servidor de conexión.
10. El sistema realiza la conexión.
11. El sistema pregunta al usuario qué dispositivo de detección de movimiento desea utilizar.
12. El usuario selecciona dispositivo.
13. En el caso de que el dispositivo sea la cámara, el sistema inicializa la cámara.
14. El sistema inicializa la partida.

Todo lo anterior se seleccionará a través de ventanas de diálogo que se irán sucediendo.

A partir de este momento, ya no hay nada preestablecido; los distintos avatares conectados al EV decidirán cuando jugar, etc.

A continuación aparecen unas imágenes tomadas de la interfaz del sistema:

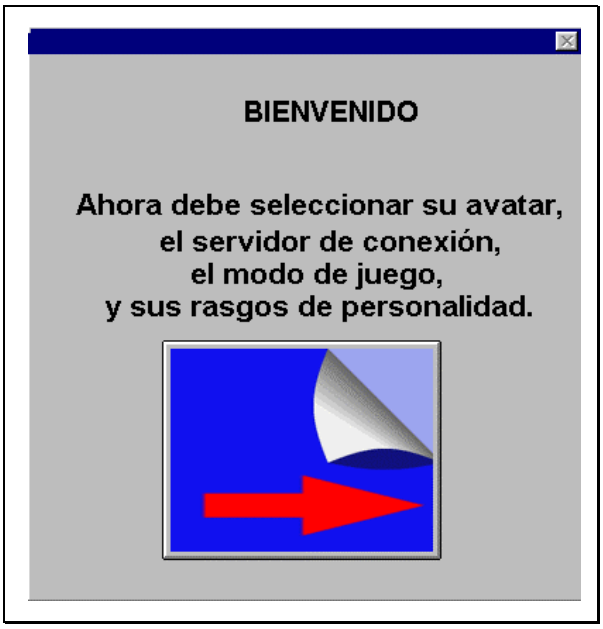

Figura 37: Pantalla de Bienvenida

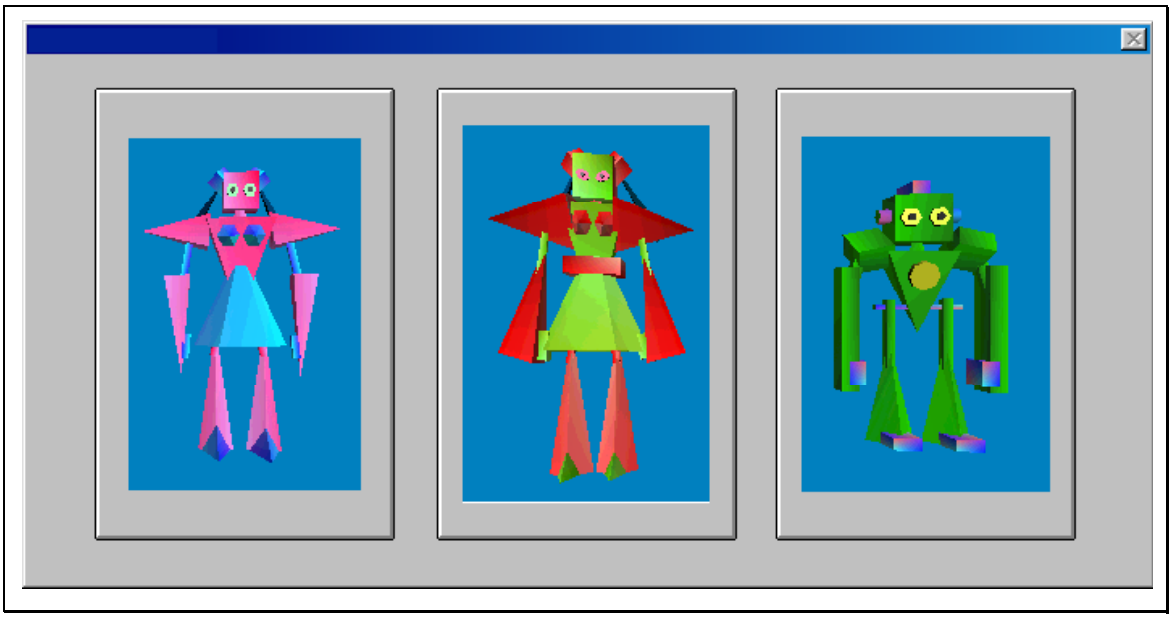

Figura 38: Selección de Avatar

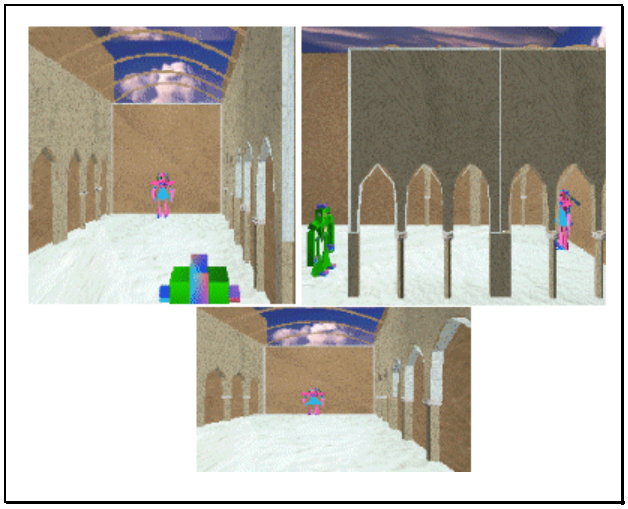

Figura 39: Selección del Punto de Vista

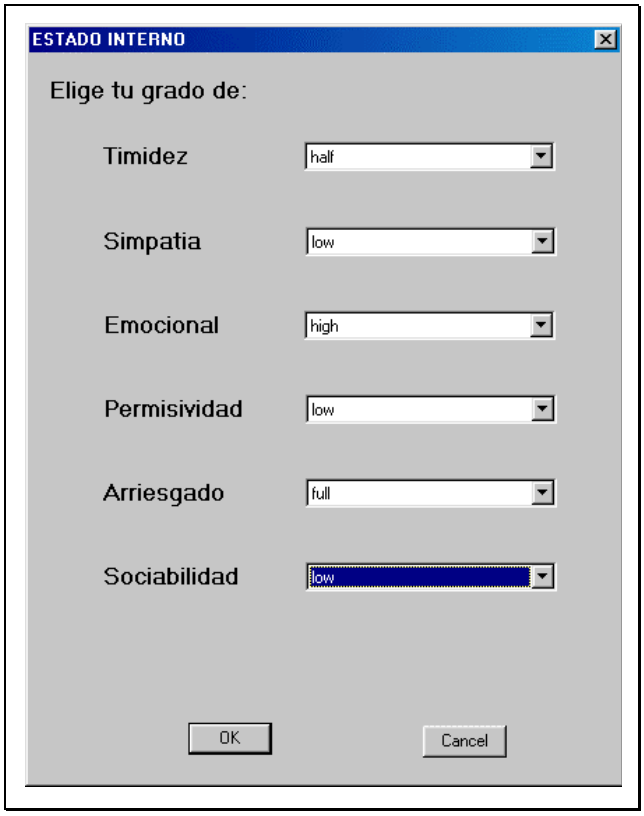

Figura 40: Selección de los rasgos de personalidad

Figura 41: Selección de Servidor de Conexión

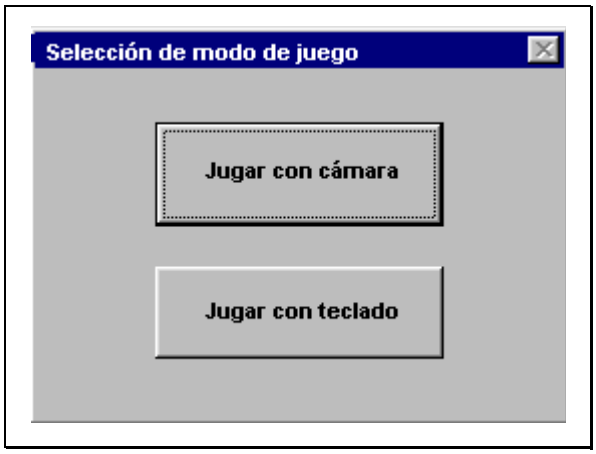

Figura 42: Selección de Dispositivo de Captura de Movimiento

Si el dispositivo seleccionado para la detección de movimiento es el teclado, bastará con pulsar las teclas que se indicaron anteriormente. Si, por el contrario, se ha seleccionado la cámara como dispositivo de captura, el procedimiento de detección es el siguiente:

1. La cámara ha detectado un cambio de posición del usuario al que corresponde el ciclo de funcionamiento.
2. Calcula el centro de masas del usuario real.
3. Convierte la coordenada del centro de masas en una posición en metros, dentro del recinto en el que se puede mover el usuario que está siendo observado por la cámara.
4. En función del valor anterior, calcula el tanto por ciento de desplazamiento a efectuar por el avatar dentro del EV.
5. La cámara le comunica al avatar que ha habido desplazamiento, y la distancia que se ha desplazado.
6. El avatar consulta el estado de humor y personalidad. Conforme a estos valores, ordena al cuerpo que se desplace de una determinada forma.

2.4. DIAGRAMAS DE INTERACCIÓN

Como siguiente paso dentro del método de Larman [Larman99] se encuentra la realización de los diagramas de interacción; concretamente, es necesario diseñar un diagrama de interacción por cada contrato de operación que se realizase en la fase de análisis.

Los diagramas de interacción pueden tomar la forma de diagramas de secuencia o de diagramas de colaboración. A pesar de expresar básicamente la misma idea, Larman aconseja la utilización de diagramas de colaboración, razón por la cual, en este primer ciclo de desarrollo, se van a utilizar los diagramas de colaboración en lugar de diagramas de secuencia.

2.4.1. ElegirAvatar (avatar)

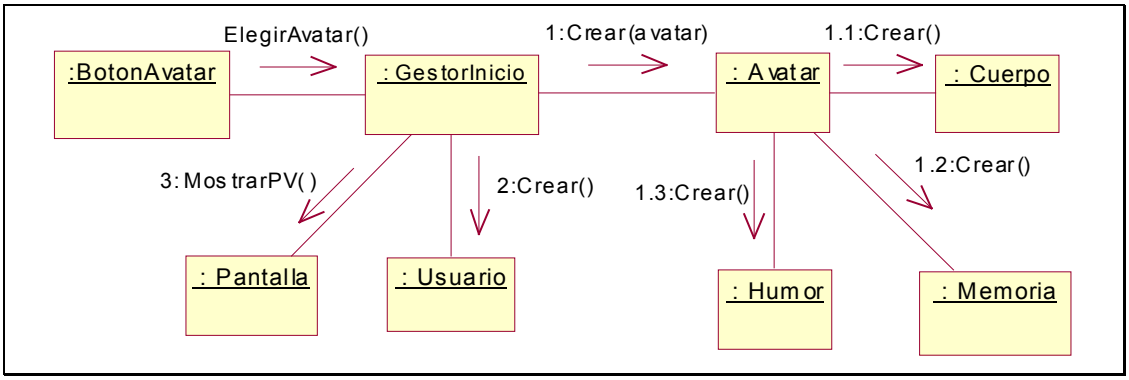

Figura 43: ElegirAvatar (avatar)

2.4.2. SeleccionarPV (pv)

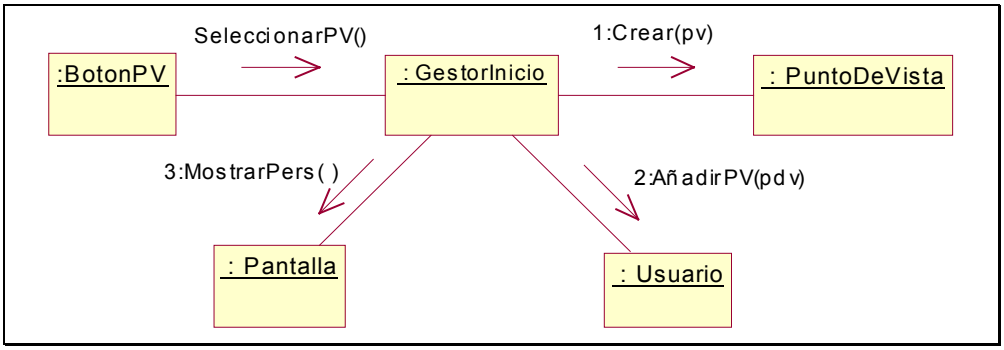

Figura 44: SeleccionarPV (pv)

2.4.3. SeleccionarPersonalidad (pers)

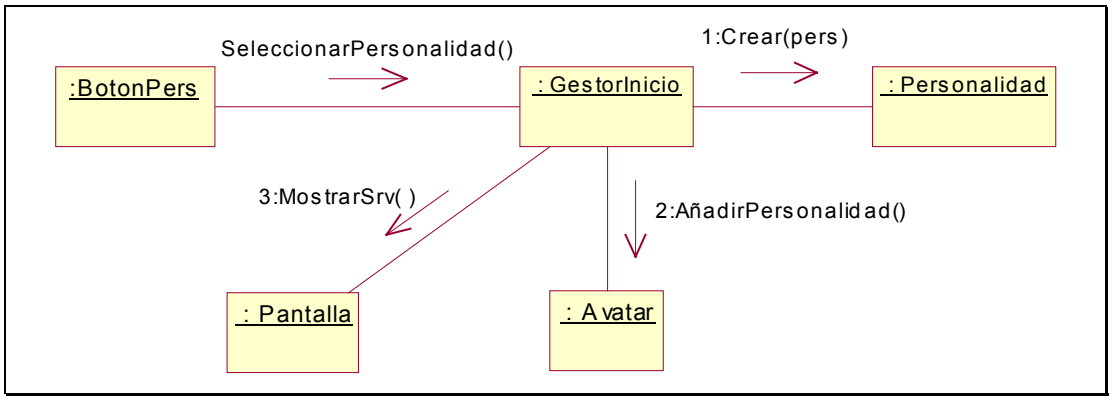

Figura 45: SeleccionarPersonalidad (pers)

2.4.4. SeleccionarServidor (nombre)

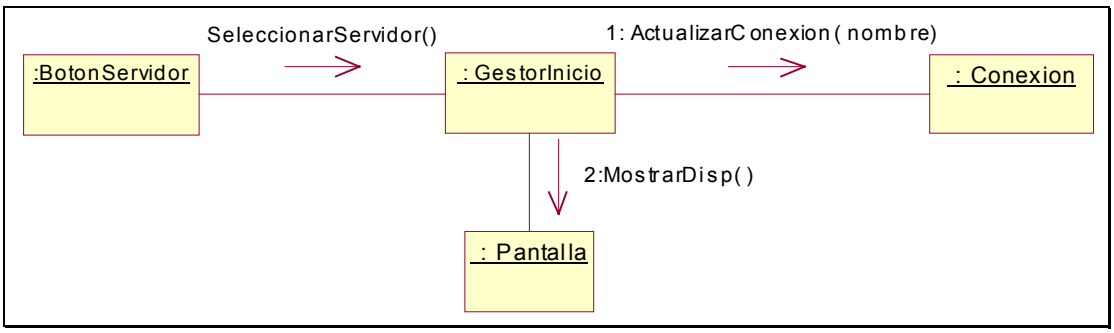

Figura 46: SeleccionarServidor (nombre)

2.4.5. SeleccionarDispositivo (disp)

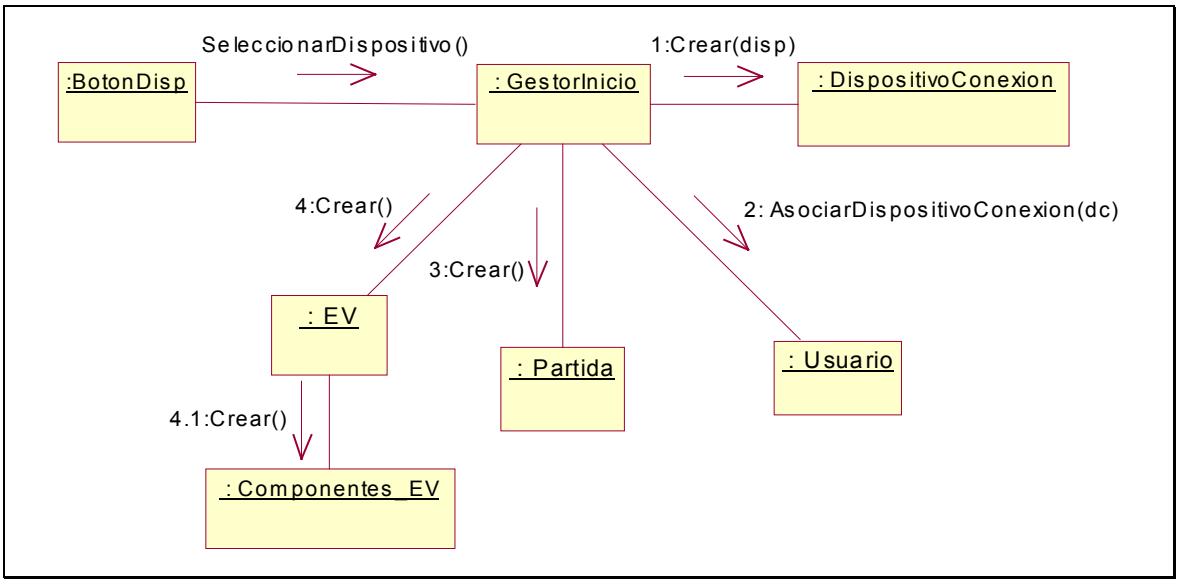

Figura 47: SeleccionarDispositivo (disp)

2.4.6. Contar ()

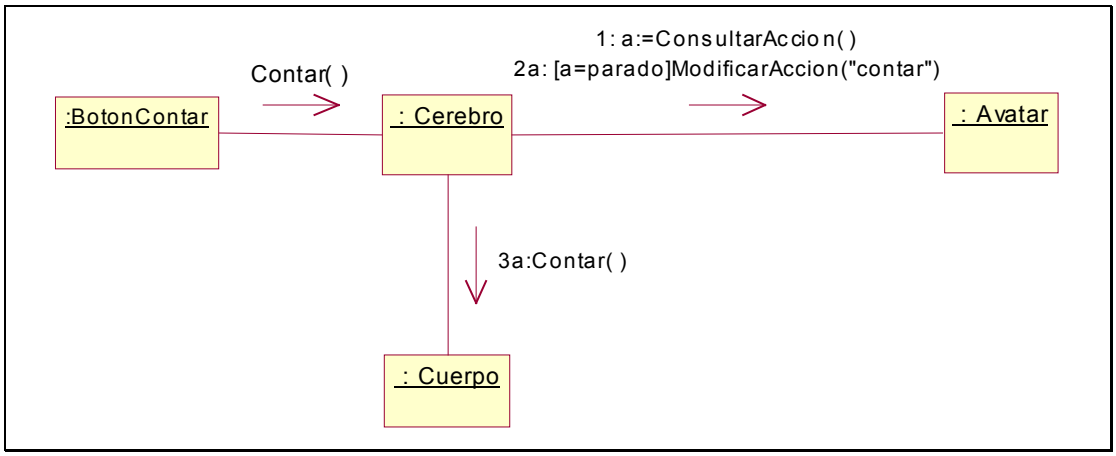

Figura 48: Contar ()

2.4.7. Saludar ()

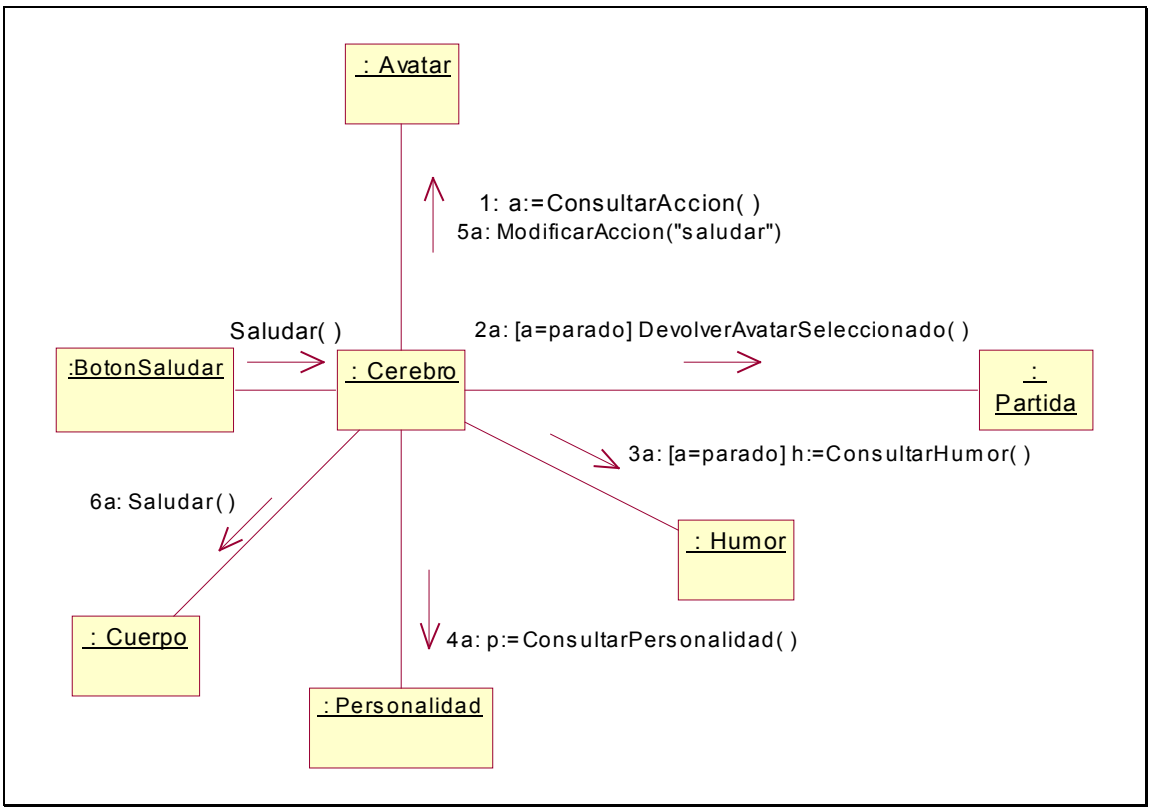

Figura 49: Saludar ()

2.4.8. SeleccionarAvatar (avatar)

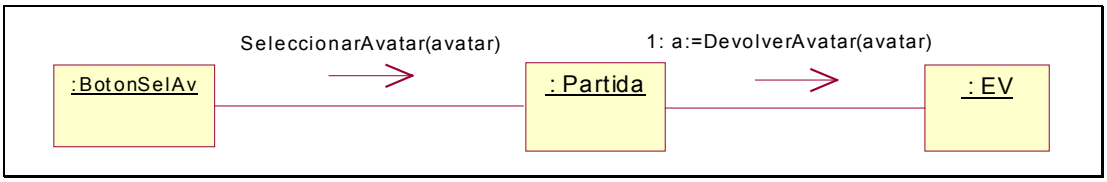

Figura 50: SeleccionarAvatar (avatar)

2.4.9. HacerReir ()

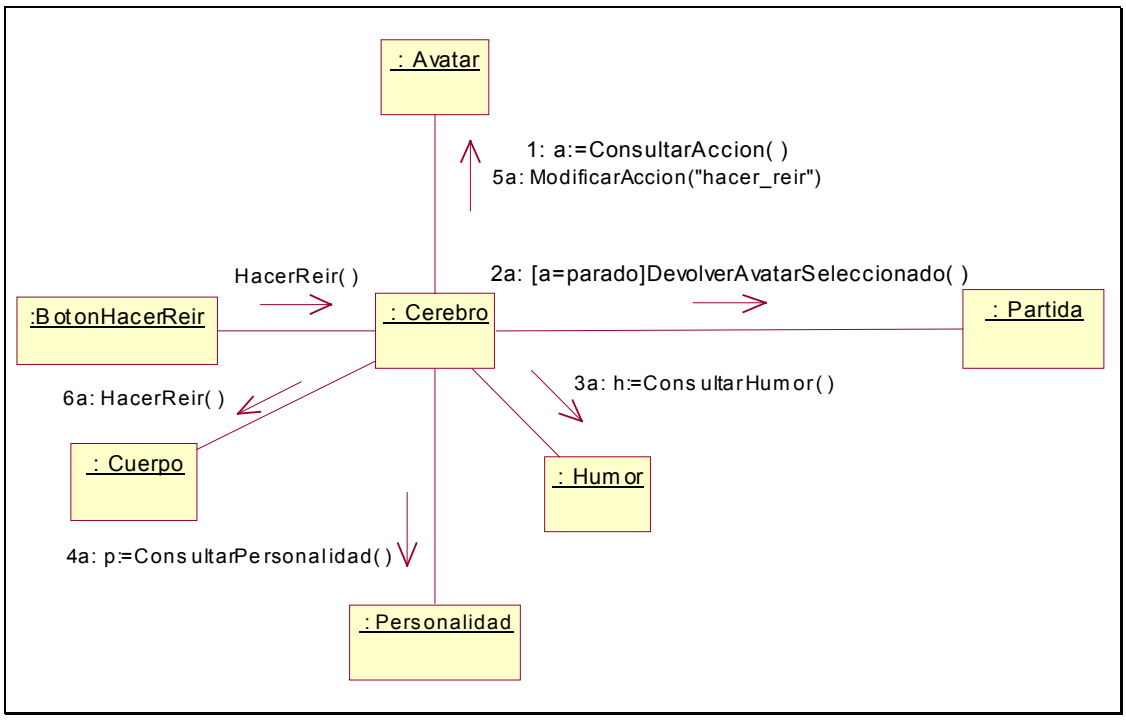

Figura 51: HacerReir ()

2.4.10. Agredir ()

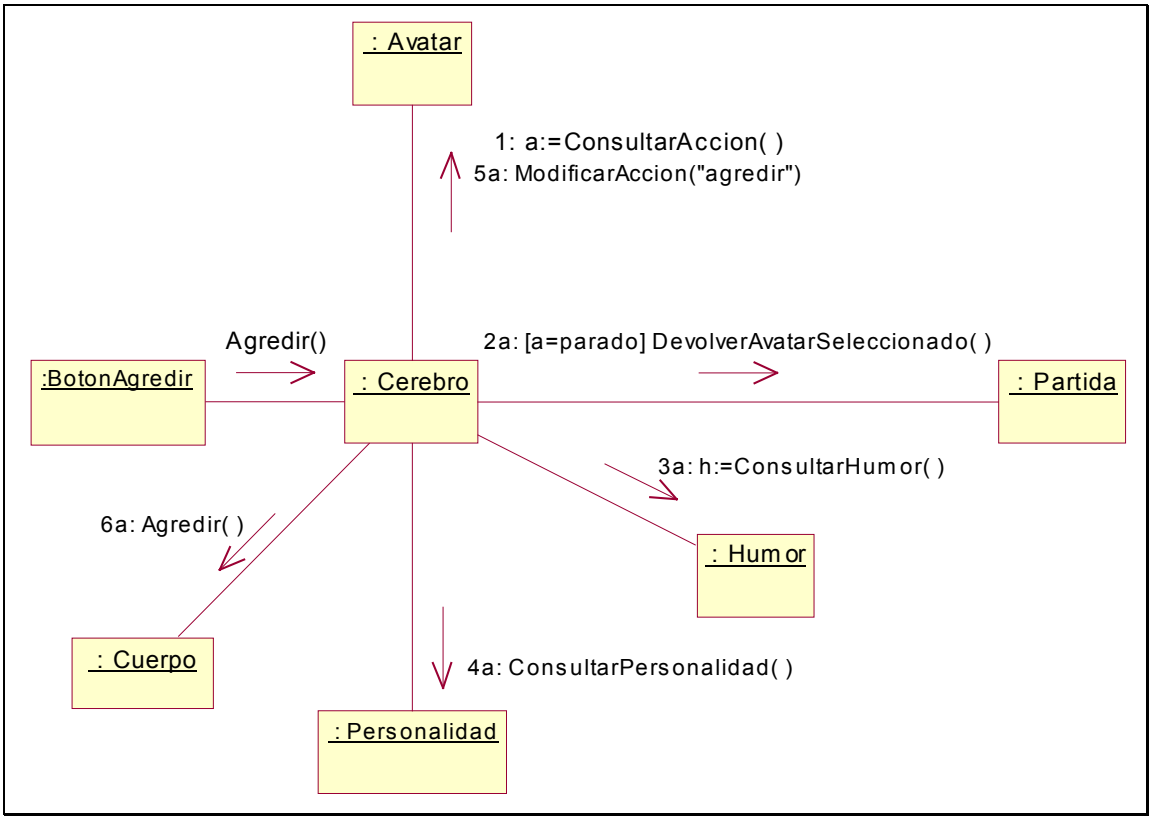

Figura 52: Agredir ()

2.4.11. Reir ()

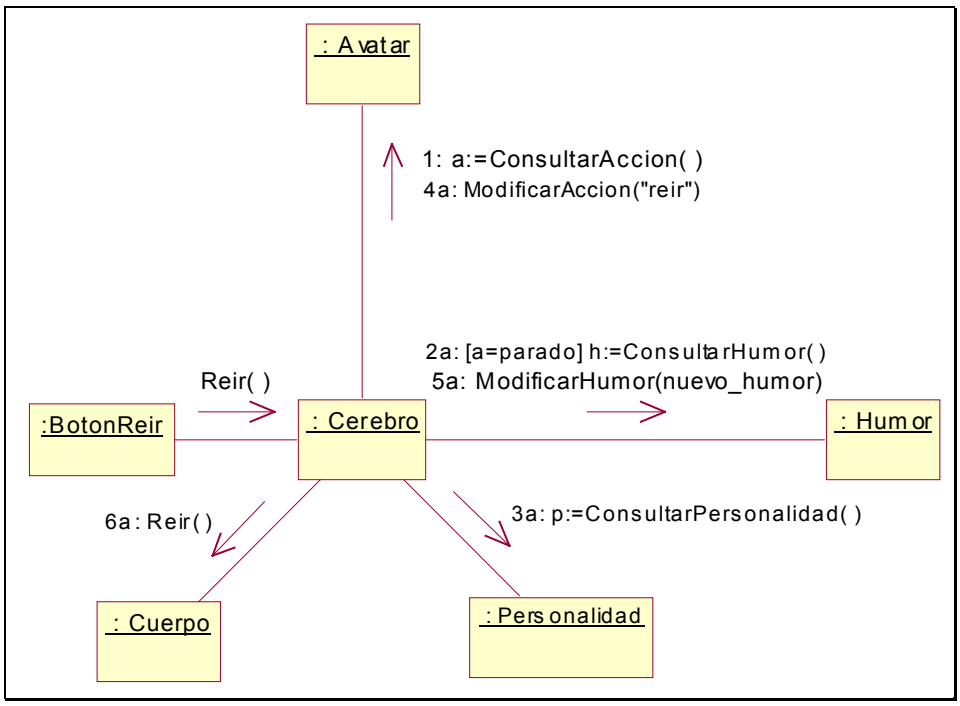

Figura 53: Reir ()

2.4.12. Llorar ()

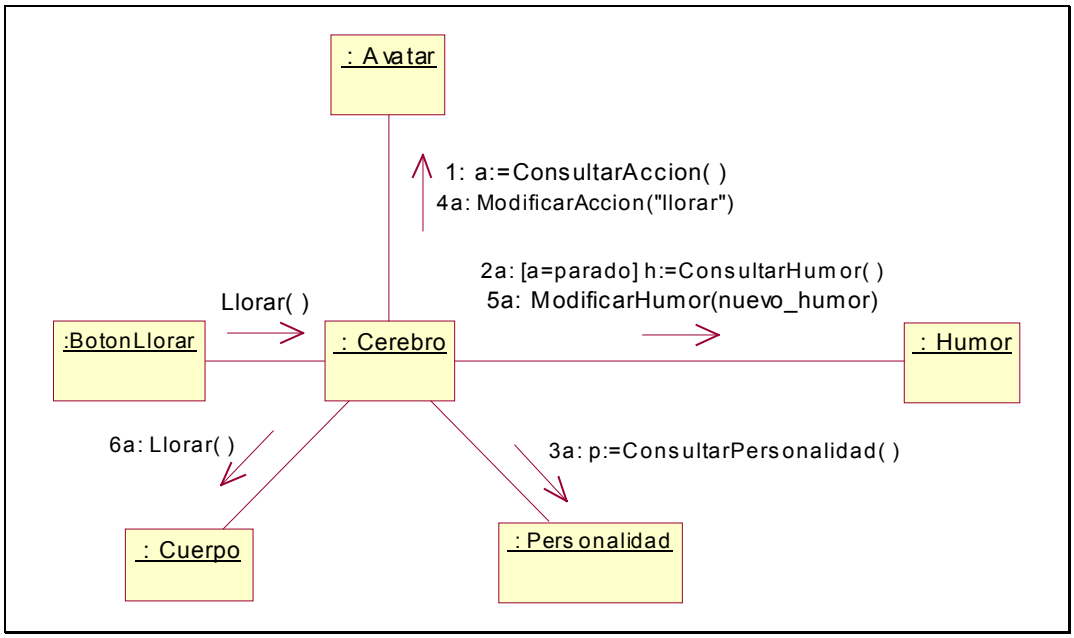

Figura 54: Llorar ()

2.4.13. PonerseTriste ()

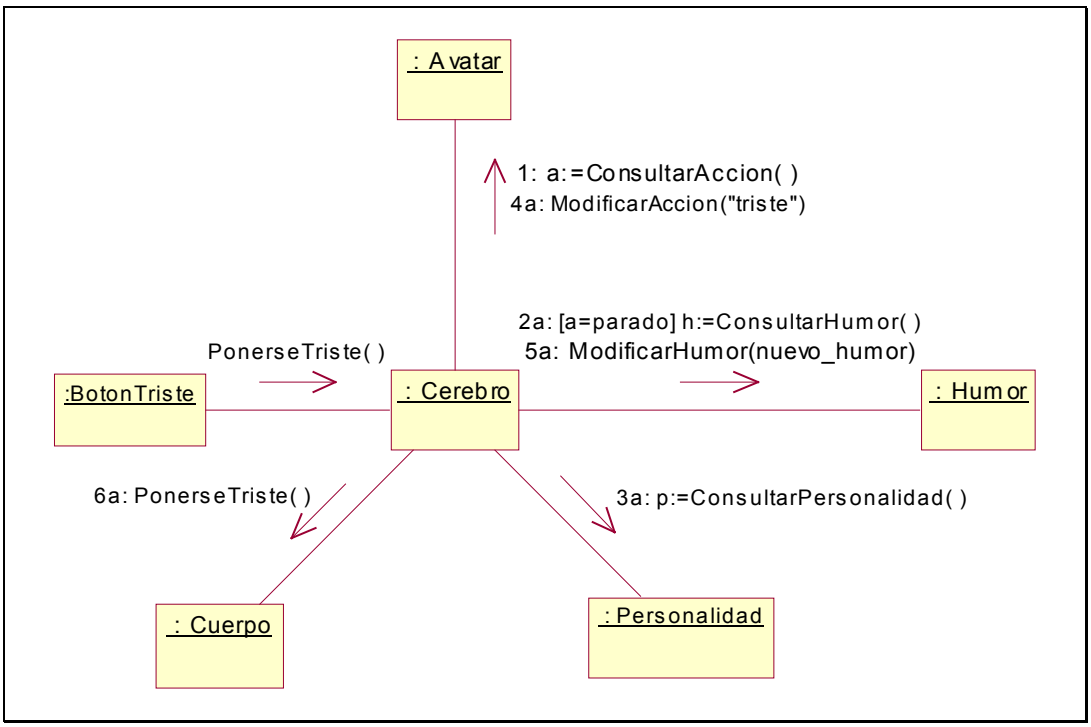

Figura 55: PonerseTriste ()

2.4.14. Enfadarse ()

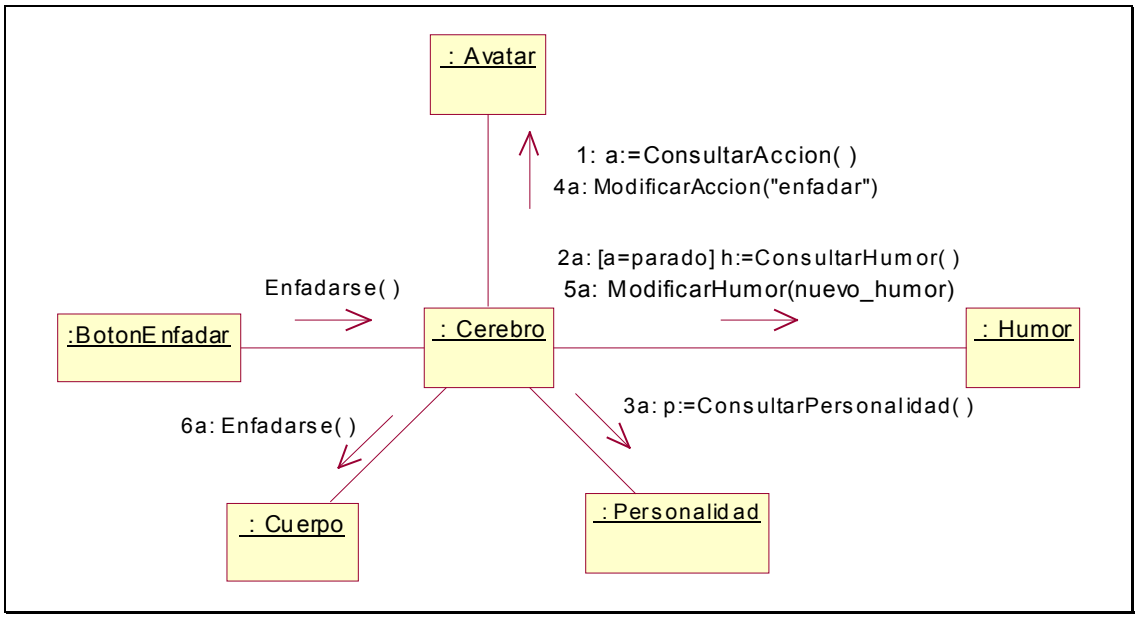

Figura 56: Enfadarse ()

2.4.15. Aburrirse ()

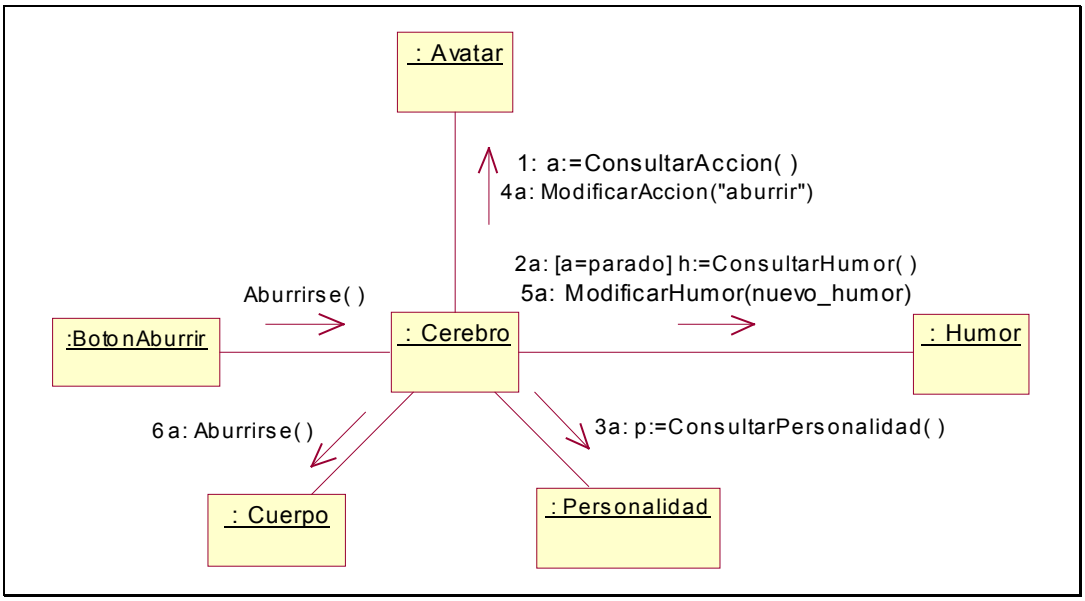

Figura 57: Aburrirse ()

2.4.16. SaltarDeAlegria ()

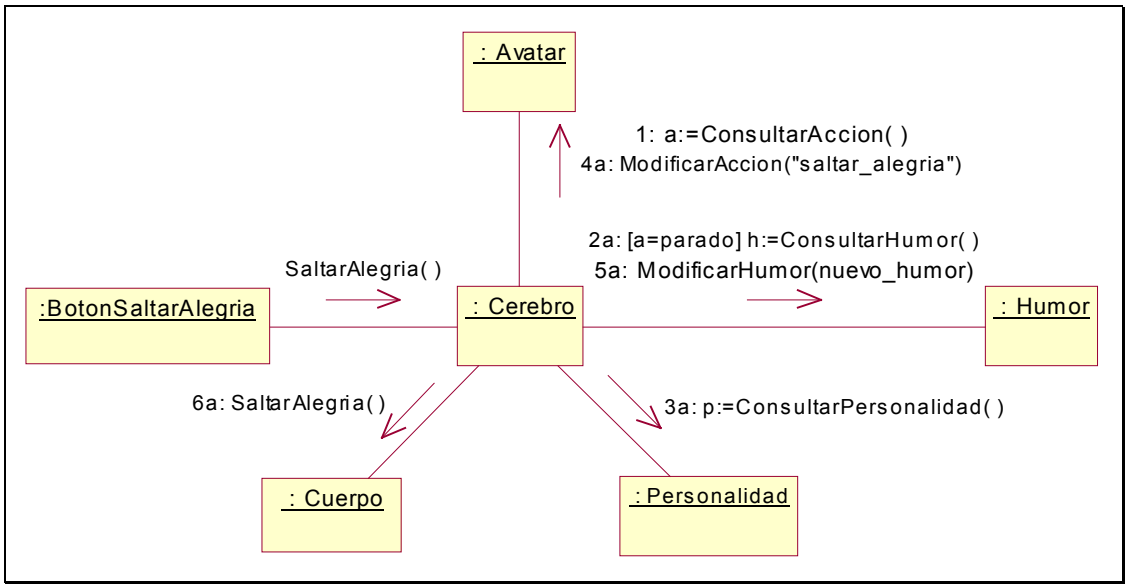

Figura 58: SaltarDeAlegria ()

2.4.17. Bailar ()

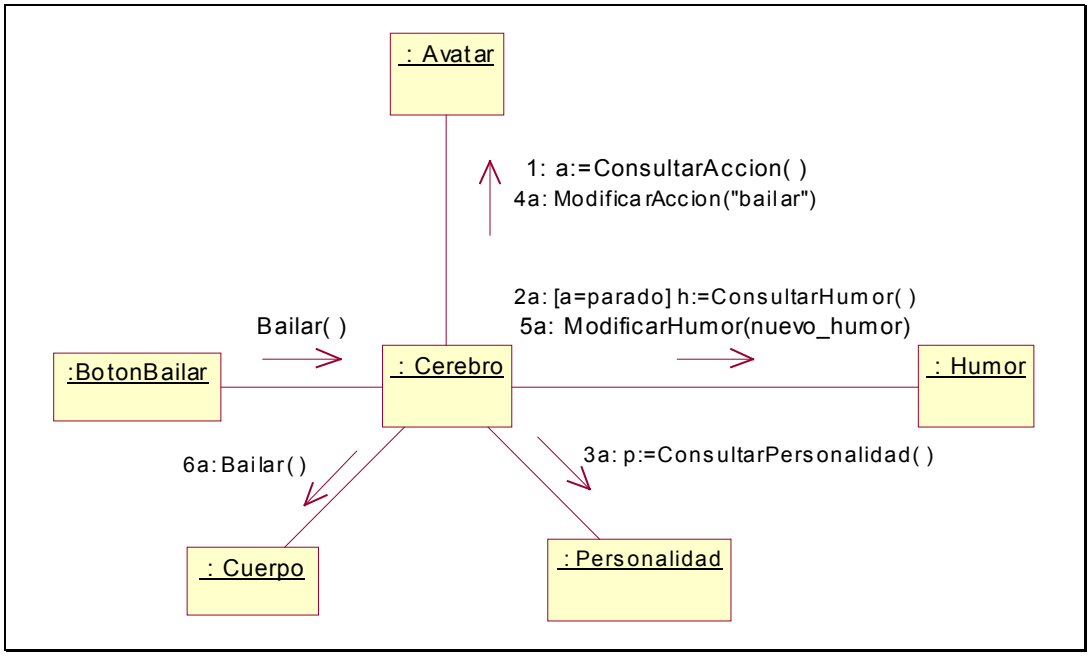

Figura 59: Bailar ()

2.4.18. Patalear ()

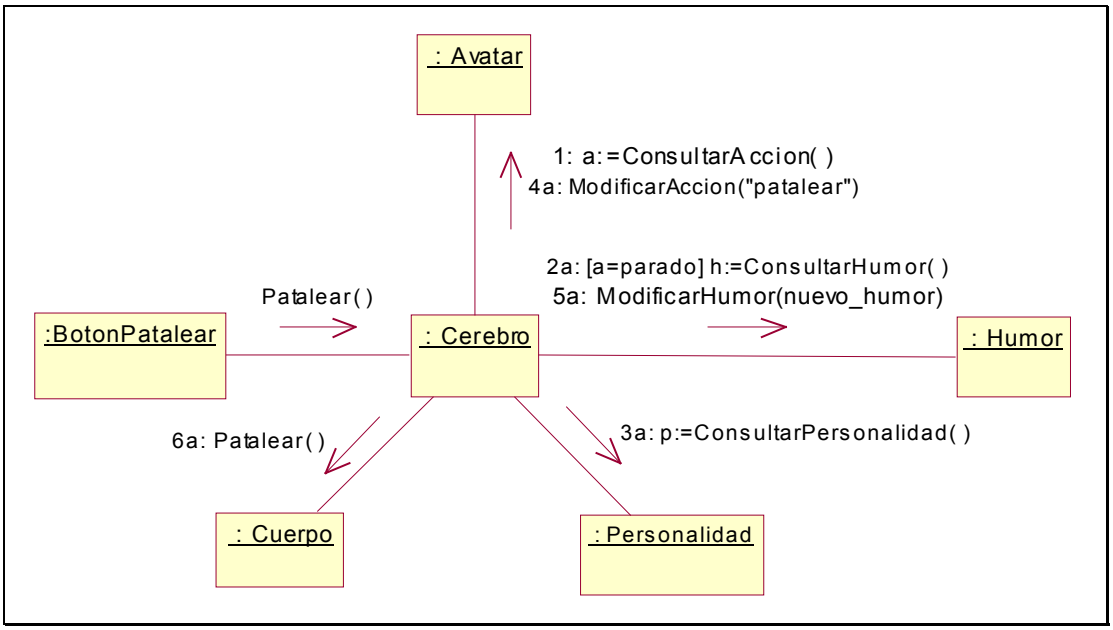

Figura 60: Patalear ()

2.4.19. ComunicarVisto ()

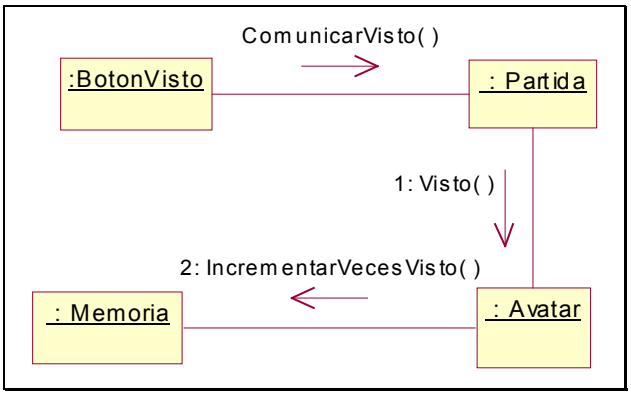

Figura 61: ComunicarVisto ()

2.4.20. VolverComienzo ()

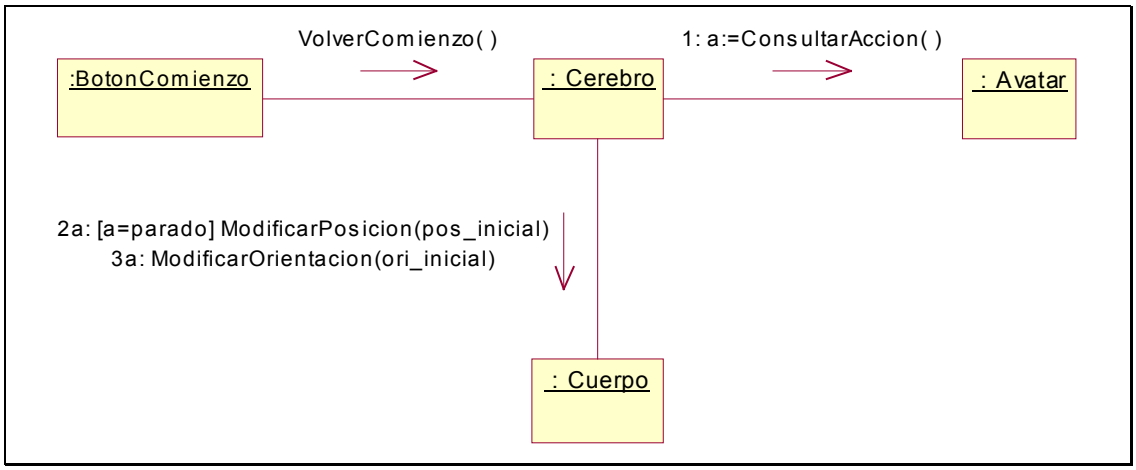

Figura 62: VolverComienzo

2.4.21. Andar (dir)

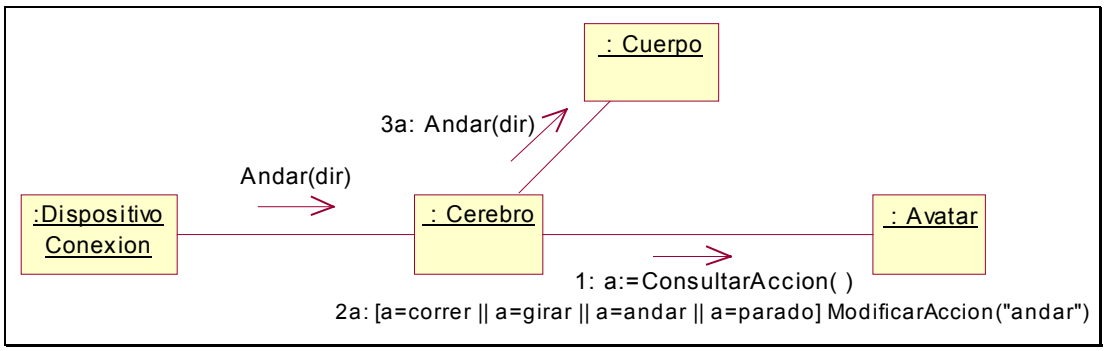

Figura 63: Andar (dir)

2.4.22. Correr ()

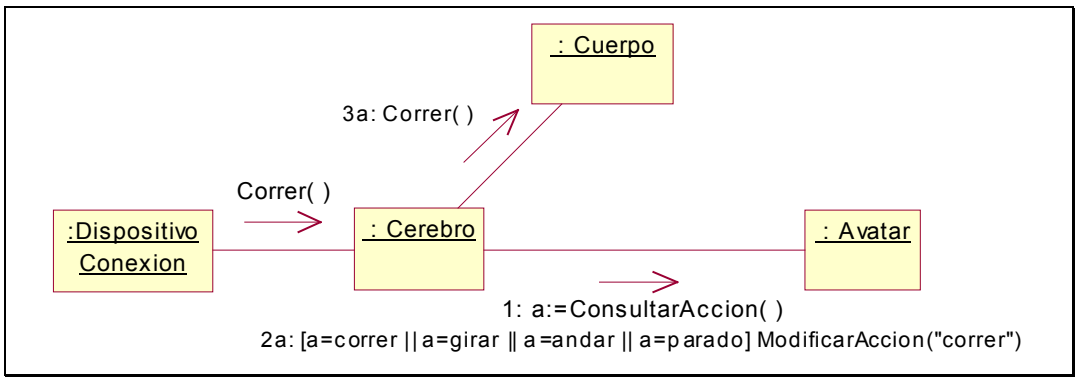

Figura 64: Correr ()

2.4.23. Girar (lado)

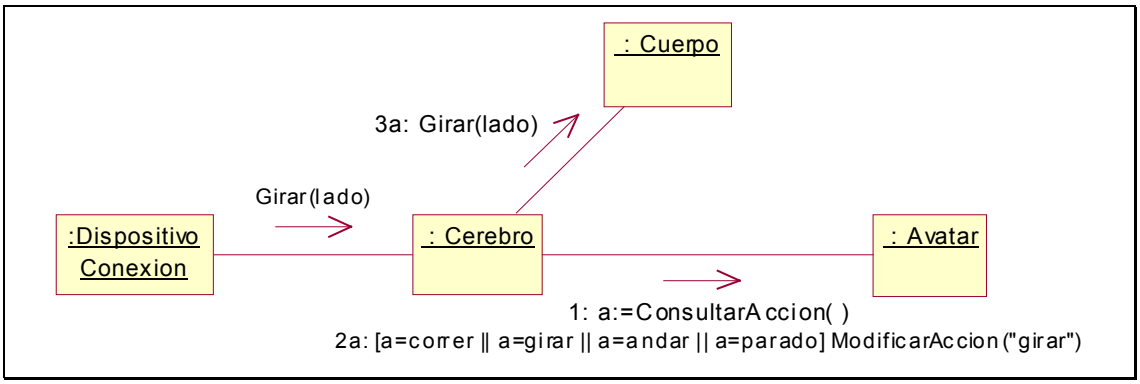

Figura 65: Girar (lado)

2.4.24. Enviar (texto)

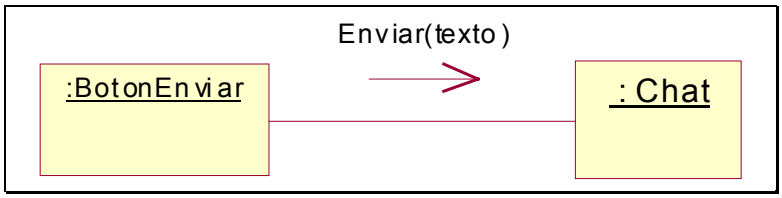

Figura 66: Enviar (texto)

2.5. DIAGRAMA DE CLASES DE DISEÑO

Este diagrama se ha realizado principalmente a partir de los diagramas de interacción y el modelo conceptual, aunque se ha intentado conseguir un diseño más modular mediante la utilización de algunos patrones de diseño.

A pesar de ello, debido a las especiales características de este tipo de sistemas, que deben funcionar en tiempo real, no se ha optimizado el diseño todo lo que se podría, ya que una de las características que también se ha intentado tener en cuenta ha sido la eficiencia, a costa, entre otros aspectos, de reducir la cohesión de las clases y aumentar el acoplamiento entre algunas de ellas.

El diseño resultante es el que se presenta a continuación:

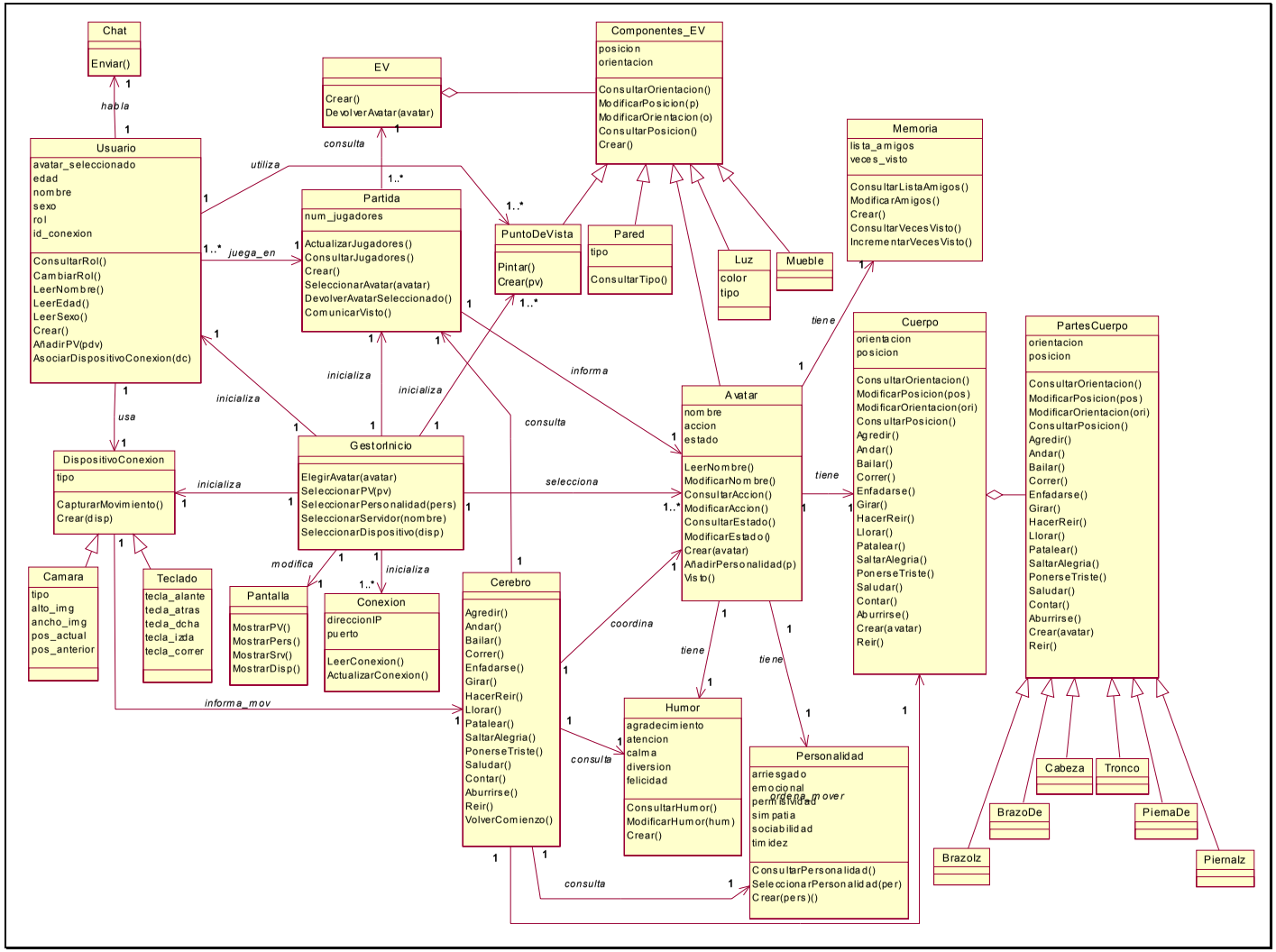

Figura 67: Diagrama de Clases de Diseño

2.6. MODELO DE DATOS

Para la realización del modelo de datos, se va a tener en cuenta que la información que se quiere almacenar es la relativa al usuario de la aplicación, junto la lista de amigos que ese usuario ha ido haciendo a lo largo de sus conexiones al EV.

También se desea poder guardar información relativa a los jugadores que se hayan conectados en cada momento, pero de esta labor se encargarán las aplicaciones seleccionadas para realizar la implementación del sistema (W2W y WorldUp).

Debido a la mínima complejidad de la información que se desea almacenar, se ha decidido que no compensa la utilización de una base de datos, razón por la cual la información que no se encarga de guardar la aplicación de desarrollo se almacenará en un archivo de texto, que tendrá el siguiente formato:

<Nombre del usuario>

<Avatar>

<Rasgos de personalidad>

<Lista de amigos>

A su vez, la lista de amigos, que se extenderá hasta el final del fichero, será un conjunto de líneas, cada una de las cuales tendrá el siguiente formato:

<Nombre del usuario><Nombre del avatar><Avatar><ID conexión>

2.7. REFINAMIENTO DE LA ARQUITECTURA DEL SISTEMA

A pesar de no contemplar explícitamente su uso, Larman sí incide en el hecho de que si se cree que ciertos elementos de UML pueden aportar alguna luz en el diseño del sistema, se pueden introducir donde se considere conveniente.

Esto sucede, por ejemplo, con los diagramas de componentes y los diagramas de despliegue, los cuales formarían parte de una etapa de refinamiento de la arquitectura del sistema, la cual no tiene un lugar fijo donde realizarse ni una forma concreta.

Por estas razones, se ha considerado conveniente aclarar, por un lado, la relación que va a existir entre los elementos *hardware* que van a conformar el sistema (diagrama

de despliegue), y por otro, la relación existente entre los elementos *software* que van a dar lugar a la representación 3D del entorno virtual (diagrama de componentes).

2.7.1. Diagrama de Componentes

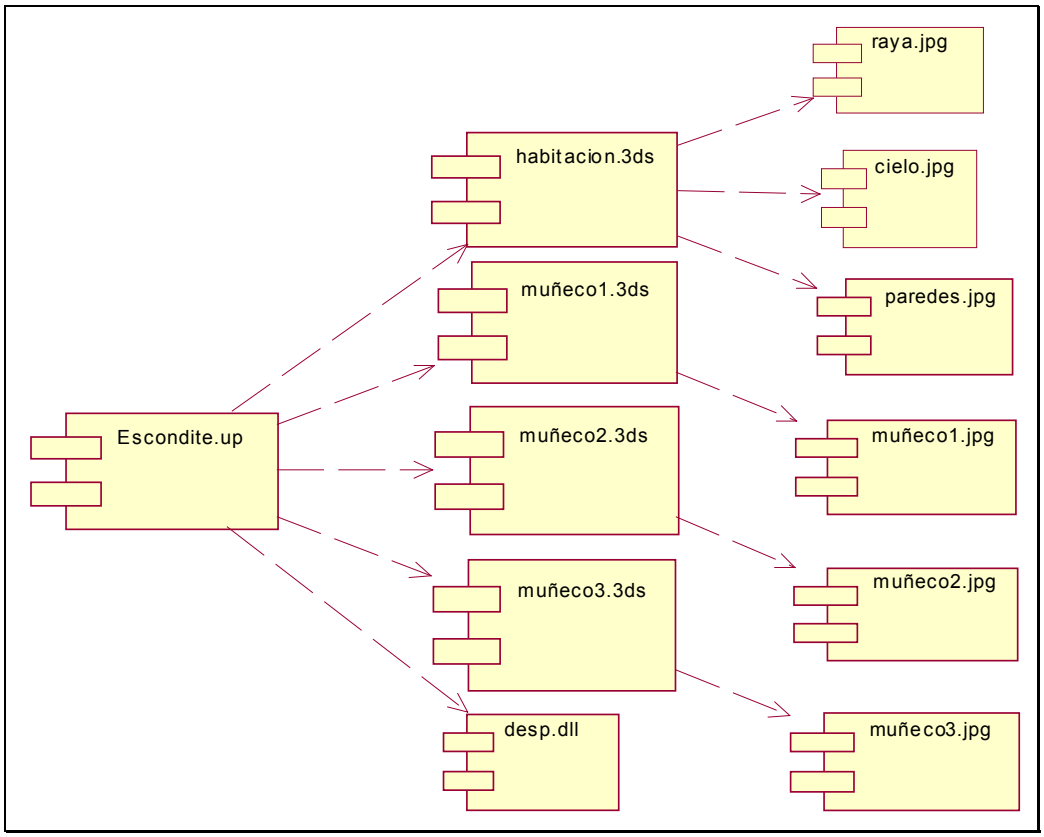

Figura 68: Diagrama de Componentes

2.7.2. Diagrama de Despliegue

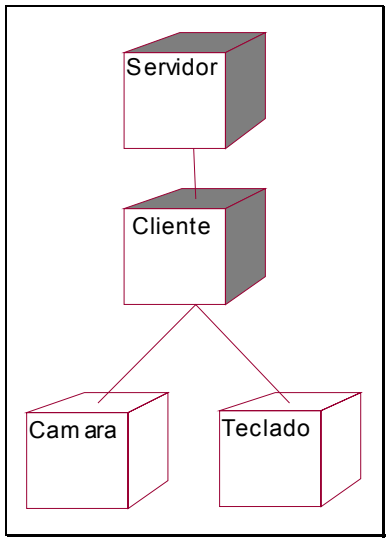

Figura 69: Diagrama de Despliegue

2.8. GLOSARIO

Seguimos ampliando el glosario con los términos que aparecen en la fase de diseño.

Término	Categoría	Descripción
Angulo_vision	*Atributo*	Ángulo máximo dentro del cual es posible ver lo que sucede alrededor del punto de visión.
Avatar	*Clase*	Representa, de manera abstracta, los avatares que puede utilizar un usuario para moverse por el EV.
Avatar.Accion	*Atributo*	Representa la acción que está realizando el avatar en un momento determinado. Sirve, entre otras cosas, para comprobar que no se realizan acciones incompatibles.
Avatar.AñadirPersonalidad (p)	*Método*	Operación para asociar a un avatar un objeto de la clase personalidad.
Avatar.ConsultarAccion ()	*Método*	Devuelve la acción que está realizando el cuerpo.
Avatar.ConsultarEstado ()	*Método*	Devuelve el estado de realización de la acción en curso.
Avatar.Crear (avatar)	*Método*	Operación para crear un nuevo avatar.
Avatar.Estado	*Atributo*	Para algunas acciones, indica el grado que se ha alcanzado en su realización.
Avatar.LeerNombre ()	*Método*	Devuelve el nombre del avatar.
Avatar.ModificarAccion ()	*Método*	Cambia la acción que está realizando el cuerpo.
Avatar.ModificarEstado ()	*Método*	Cambia el estado de la acción que se está realizando actualmente.
Avatar.ModificarNombre ()	*Método*	Cambia el nombre del avatar.
Avatar.Nombre	*Atributo*	Nombre que el usuario le da a su avatar.
Avatar.Visto ()	*Método*	Operación para comunicarle al avatar que ha sido visto moviéndose.
BrazoDe	*Clase*	Clase que representa el brazo derecho de los avatares.
BrazoIz	*Clase*	Clase que representa el brazo izquierdo de los avatares.
Cabeza	*Clase*	Clase que representa la cabeza de los avatares.
Camara	*Clase*	Dispositivo de conexión para mover al avatar por el EV.
Camara.Alto_img	*Atributo*	Alto de la imagen que captura.
Camara.Ancho_img	*Atributo*	Ancho de la imagen que captura.

Camara.Pos_actual	*Atributo*	Posición en la que ha captado al usuario.
Camara.Pos_anterior	*Atributo*	Posición anterior en la que captó al usuario.
Camara.Tipo	*Atributo*	Tipo de la cámara utilizada.
Cerebro	*Clase*	Esta clase se encarga de gestionar cómo debe realizar el avatar las acciones.
Cerebro.Aburrirse ()	*Método*	Realiza el movimiento para mostrar aburrimiento.
Cerebro.Agredir ()	*Método*	Realiza el movimiento de agresión a otro avatar.
Cerebro.Andar ()	*Método*	Realiza el movimiento de andar para desplazarse por el EV.
Cerebro.Bailar ()	*Método*	Realiza el movimiento de bailar.
Cerebro.Contar ()	*Método*	Realiza el movimiento de volverse a la pared y luego a los jugadores.
Cerebro.Correr ()	*Método*	Realiza la acción de correr para desplazarse por el EV.
Cerebro.Enfadarse ()	*Método*	Realiza el movimiento para mostrar enfado.
Cerebro.Girar ()	*Método*	Realiza la acción de girar el cuerpo.
Cerebro.HacerReir ()	*Método*	Realiza el movimiento para hacer reír a otro avatar.
Cerebro.Llorar ()	*Método*	Realiza el movimiento de llorar.
Cerebro.Patalear ()	*Método*	Realiza el movimiento de patalear.
Cerebro.PonerseTriste ()	*Método*	Realiza el movimiento para expresar tristeza.
Cerebro.Reir ()	*Método*	Operación para que el avatar se pueda reír.
Cerebro.SaltarAlegria ()	*Método*	Realiza el movimiento para saltar de alegría.
Cerebro.Saludar ()	*Método*	Realiza el movimiento para saludar.
Cerebro.VolverComienzo ()	*Método*	Operación para indicarle al avatar que vuelva a la línea de comienzo.
Chat	*Clase*	Componente que utilizan los usuarios para comunicarse entre ellos.
Chat.Enviar ()	*Método*	Envía un texto escrito por un usuario al resto de los usuarios.
Componentes_EV	*Clase*	Elementos que puede haber dentro del ev: muebles, puertas, avatares...
Componentes_EV.ConsultarOrientacion ()	*Método*	Devuelve la orientación del componente dentro del EV.
Componentes_EV .ConsultarPosicion ()	*Método*	Devuelve la posición del componente dentro del EV.

Componentes_EV .Crear ()	*Método*	Operación para crear nuevos componentes para el EV.
Componentes_EV.ModificarOrientacion (o)	*Método*	Cambia la orientación del componente dentro del EV.
Componentes_EV .ModificarPosicion (p)	*Método*	Cambia la posición del componente dentro del EV.
Componentes_EV.Orientacion	*Atributo*	Indica la orientación del componente dentro del EV.
Componentes_EV.Posicion	*Atributo*	Establece la posición del componente dentro del EV.
Conexión	*Clase*	Representa la conexión que se establece con el servidor de mundos virtuales.
Conexión.ActualizarConexion ()	*Método*	Cambia los valores de la conexión actual.
Conexión.DireccionIP	*Atributo*	Dirección IP del servidor.
Conexión.LeerConexion ()	*Método*	Devuelve el valor de la conexión que se encuentra actualmente activa.
Conexión.Puerto	*Atributo*	Puerto por el que se realiza la conexión.
consulta	*Asociación*	Una Partida consulta lo que sucede en el EV.
consulta	*Asociación*	El Cerebro debe consultar los rasgos de Humor para saber cómo realiza ciertas acciones. También debe modificar estos rasgos como respuesta a los sucesos acaecidos dentro del EV.
consulta	*Asociación*	El Cerebro necesita consultar los rasgos de Personalidad para saber cómo realizar ciertas acciones.
consulta	*Asociación*	El Cerebro consulta los datos guardados por la Partida para poder decidir cómo realizar algunas acciones.
consulta	*Asociación*	El Cerebro consulta los Sentidos para determinar si sucede algo en el EV.
consulta	*Asociación*	El Cerebro consulta la Memoria para determinar si debe hacer algo con los datos de los que dispone.
consulta	*Asociación*	Los Sentidos Consultan los Componentes del EV para determinar si pueden interactuar con ellos.
coordina	*Asociación*	Un Avatar tiene un Cerebro que se encarga de coordinar sus acciones.
Cuerpo	*Clase*	Cuerpo del avatar que representa al usuario dentro del EV.
Cuerpo.Aburrirse ()	*Método*	Realiza el movimiento para mostrar aburrimiento.
Cuerpo.Agredir ()	*Método*	Realiza el movimiento de agresión a otro avatar.

Cuerpo.Andar ()	*Método*	Realiza el movimiento de andar para desplazarse por el EV.
Cuerpo.Bailar ()	*Método*	Realiza el movimiento de bailar.
Cuerpo.ConsultarOrientacion ()	*Método*	Devuelve la orientación del cuerpo dentro del EV.
Cuerpo.ConsultarPosicion ()	*Método*	Devuelve la posición del cuerpo dentro del EV.
Cuerpo.Contar ()	*Método*	Realiza el movimiento de volverse a la pared y luego a los jugadores.
Cuerpo.Correr ()	*Método*	Realiza la acción de correr para desplazarse por el EV.
Cuerpo.Crear (avatar)	*Método*	Operación para crear el cuerpo del avatar elegido.
Cuerpo.Enfadarse ()	*Método*	Realiza el movimiento para mostrar enfado.
Cuerpo.Girar ()	*Método*	Realiza la acción de girar el cuerpo.
Cuerpo.Hacerreir ()	*Método*	Realiza el movimiento para hacer reír a otro avatar.
Cuerpo.Llorar ()	*Método*	Realiza el movimiento de llorar.
Cuerpo.ModificarOrientacion (o)	*Método*	Cambia la orientación del cuerpo dentro del EV.
Cuerpo.ModificarPosicion (p)	*Método*	Cambia la posición del cuerpo dentro del EV.
Cuerpo.Orientacion	*Atributo*	Indica la orientación del cuerpo dentro del EV.
Cuerpo.Patalear ()	*Método*	Realiza el movimiento de patalear.
Cuerpo.PonerseTriste ()	*Método*	Realiza el movimiento para expresar tristeza.
Cuerpo.Posicion	*Atributo*	Guarda la situación del cuerpo en el EV.
Cuerpo.Reir ()	*Método*	Operación para realizar la acción de reír.
Cuerpo.SaltarAlegria ()	*Método*	Realiza el movimiento para saltar de alegría.
Cuerpo.Saludar ()	*Método*	Realiza el movimiento para saludar.
DispositivoConexion	*Clase*	Dispositivo que utiliza el usuario para conectarse al EV y desplazar a su avatar.
DispositivoConexion.CapturarMovimiento ()	*Método*	Sirve para detectar si el usuario ha ordenado a su avatar que se mueva, bien a través de una pulsación de teclado, un movimiento detectado por la cámara, ...
DispositivoConexion.Crear (disp)	*Método*	Operación para crear un dispositivo de conexión.
DispositivoConexion.Tipo	*Atributo*	Tipo de dispositivo que utiliza el usuario para conectarse al EV y controlar su avatar: teclado o cámara.

EV	*Clase*	Entorno Virtual en el que se van a desarrollar las actividades.
EV.Crear ()	*Método*	Operación para crear un nuevo EV.
EV.DevolverAvatar (avatar)	*Método*	Devuelve el avatar cuyo identificador es el pasado como parámetro.
GestorInicio	*Clase*	Clase que gestiona la conexión del usuario al EV.
GestorInicio.ElegirAvatar (avatar)	*Método*	Operación para elegir un avatar de entre los disponibles.
GestorInicio.SeleccionarDispositivo(dis)	*Método*	Operación para seleccionar el dispositivo de conexión que utilizará el usuario para moverse por el EV.
GestorInicio.SeleccionarPersonalidad(p)	*Método*	Operación para crear la personalidad del avatar.
GestorInicio.SeleccionarPV (pv)	*Método*	Operación para seleccionar un punto de vista de entre los disponibles.
GestorInicio.SeleccionarServidor (nom)	*Método*	Operación para conectarse al servidor de Mundos Virtuales.
habla_con	*Asociación*	Un Usuario habla con otros usuarios a través de un Chat.
Humor	*Clase*	Representa los rasgos de humor de un avatar, los cuales determinan en cierta medida las reacciones que un avatar tendrá ante los sucesos del EV.
Humor.Agradecimiento	*Atributo*	Grado de agradecimiento de un avatar. Varía entre agradecido y enfadado.
Humor.Atencion	*Atributo*	Grado de atención de un avatar. Varía entre muy atento y poco atento.
Humor.Calma	*Atributo*	Grado de nerviosismo de un avatar. Varía entre calmado y excitado.
Humor.ConsultarHumor ()	*Método*	Devuelve los rasgos del humor del avatar.
Humor.Crear ()	*Método*	Crea el humor del avatar y lo rellena con valores neutros.
Humor.Diversion	*Atributo*	Grado de diversión de un avatar. Varía entre divertido y aburrido.
Humor.Felicidad	*Atributo*	Grado de felicidad de un avatar. Varía entre feliz y triste.
Humor.ModificarHumor (hum)	*Método*	Cambia los valores del humor del avatar.
informa	*Asociación*	La Partida informa al Avatar de ciertas situaciones que se pueden producir, como que sea visto moviéndose.
informa	*Asociación*	El Dispositivo de conexión informa al Cerebro de los movimientos sucedidos en el mundo real para que los traslade al EV.
inicializa	*Asociación*	El Gestor de inicio está encargado de crear Usuarios e inicializarlos.

inicializa	*Asociación*	El Gestor de inicio está encargado de crear un Punto de vista al entrar en el EV.
inicializa	*Asociación*	El Gestor de inicio está encargado de inicializar la Conexión al servidor de mundos virtuales.
inicializa	*Asociación*	El Gestor de inicio está encargado de crear e inicializar los Dispositivos de conexión.
inicializa	*Asociación*	El Gestor de inicio está encargado de crear las Partidas.
juega_en	*Asociación*	Un Usuario juega en una Partida dentro de un EV.
Luz	*Clase*	Fuentes de luz para iluminar el EV, con el objetivo de crear distintas atmósferas.
Luz.Color	*Atributo*	Color de la luz.
Luz.Tipo	*Atributo*	Indica si la luz es ambiental, dirigida...
Memoria	*Clase*	Permite que el avatar 'recuerde' cosas de interés de otras ocasiones en las que se conectó al EV, lo que le capacita para reaccionar de distinta forma, por ejemplo, al ver a un avatar al que ya conoce.
Memoria.ConsultarListaAmigos ()	*Método*	Devuelve la lista de amigos del avatar.
Memoria.ConsultarVecesVisto ()	*Método*	Devuelve el número de veces que el jugador ha sido visto moviéndose a lo largo del juego.
Memoria.Crear ()	*Método*	Operación para crear la memoria del avatar.
Memoria.IncrementarVecesVisto ()	*Método*	Cuando un jugador es visto moviéndose, con esta operación se incrementa el número de veces que lo han visto, de manera que se pueda reaccionar de alguna manera si le ven moverse muchas veces.
Memoria.Lista_amigos	*Atributo*	Lista de avatares a los que se ha conocido en otras partidas en el EV.
Memoria.ModificarAmigos ()	*Método*	Añade nuevos amigos en la lista de amigos.
Memoria.Veces_visto	*Atributo*	Número de veces que un avatar ha sido visto moviéndose.
modifica	*Asociación*	El Gestor de inicio modifica la Pantalla para ir mostrando las distintas elecciones que tiene que realizar el usuario al conectarse.
Mueble	*Clase*	Mobiliario de un EV.
ordena_mover	*Asociación*	El Cerebro ordena al Cuerpo que realice ciertas acciones por el EV.
Pantalla	*Clase*	Pantalla de la aplicación.
Pantalla.MostrarDisp ()	*Método*	Muestra la pantalla de selección de dispositivo de conexión.

Pantalla.MostrarPers ()	*Método*	Muestra la pantalla de selección de personalidad.
Pantalla.MostrarPV ()	*Método*	Muestra la pantalla de selección de punto de vista.
Pantalla.MostrarSrv ()	*Método*	Mostrar la pantalla de selección de servidor de conexión.
Pared	*Clase*	Sirven para delimitar las estancias que conforman el EV.
Pared.ConsultarTipo ()	*Método*	Devuelve el tipo de la pared.
Pared.Tipo	*Atributo*	Puesto que puede ser necesario saber si una determinada pared tiene alguna función, este atributo sirve para identificarla.
PartesCuerpo	*Clase*	Clase que hace de interfaz entre el cuerpo del avatar y las partes del cuerpo concretas, de manera que todas puedan ser tratadas de la misma manera.
PartesCuerpo.Aburrirse ()	*Método*	Realiza el movimiento para mostrar aburrimiento.
PartesCuerpo.Agredir ()	*Método*	Realiza el movimiento de agresión a otro avatar.
PartesCuerpo.Andar ()	*Método*	Realiza el movimiento de andar para desplazarse por el EV.
PartesCuerpo.Bailar ()	*Método*	Realiza el movimiento de bailar.
PartesCuerpo.ConsultarOrientacion ()	*Método*	Devuelve la orientación de la parte del cuerpo en relación con éste.
PartesCuerpo.ConsultarPosicion ()	*Método*	Devuelve la posición de la parte del cuerpo en relación a éste.
PartesCuerpo.Contar ()	*Método*	Realiza el movimiento de volverse a la pared y luego a los jugadores.
PartesCuerpo.Correr ()	*Método*	Realiza la acción de correr para desplazarse por el EV.
PartesCuerpo.Crear (avatar)	*Método*	Operación para crear la parte del cuerpo del avatar elegido.
PartesCuerpo.Enfadarse ()	*Método*	Realiza el movimiento para mostrar enfado.
PartesCuerpo.Girar ()	*Método*	Realiza la acción de girar el cuerpo.
PartesCuerpo.HacerReir ()	*Método*	Realiza el movimiento para hacer reír a otro avatar.
PartesCuerpo.Llorar ()	*Método*	Realiza el movimiento de llorar.
PartesCuerpo.ModificarOrientacion (ori)	*Método*	Cambia la orientación de la parte del cuerpo.
PartesCuerpo.ModificarPosicion (pos)	*Método*	Cambia la posición de la parte del cuerpo.
PartesCuerpo.Orientacion	*Atributo*	Indica la orientación de la parte del cuerpo en relación con el cuerpo.
PartesCuerpo.Patalear ()	*Método*	Realiza el movimiento de patalear.
PartesCuerpo.PonerseTriste ()	*Método*	Realiza el movimiento para expresar tristeza.

PartesCuerpo.Posicion	*Atributo*	Guarda la situación de la parte del cuerpo en relación con el cuerpo.
PartesCuerpo.Reir ()	*Método*	Operación para realizar la acción de reír.
PartesCuerpo.SaltarAlegria ()	*Método*	Realiza el movimiento para saltar de alegría.
PartesCuerpo.Saludar ()	*Método*	Realiza el movimiento para saludar.
Partida	*Clase*	Tendrá una única instancia que servirá para guardar datos generales del desarrollo del juego.
Partida.ActualizarJugadores ()	*Método*	Actualiza el número de jugadores que hay en una partida cuando se conecta un nuevo usuario o cuando un usuario abandona el juego.
Partida.ComunicarVisto ()	*Método*	Se le comunica al avatar seleccionado que se le ha visto moviéndose.
Partida.ConsultarJugadores ()	*Método*	Devuelve el número de jugadores que hay actualmente en una partida.
Partida.Crear ()	*Método*	Operación para crear un objeto de la clase Partida.
Partida.DevolverAvatarSeleccionado()	*Método*	Devuelve el avatar que se encuentra seleccionado para realizar alguna acción sobre él.
Partida.Num_jugadores	*Atributo*	Número de jugadores que hay en una partida.
Partida.SeleccionarAvatar (avatar)	*Método*	Operación para seleccionar un avatar del EV con el que se realizará una acción.
Personalidad	*Clase*	Representa los rasgos de personalidad del avatar, los cuales determinan en cierta forma las reacciones y el comportamiento del avatar dentro del EV.
Personalidad.Arriesgado	*Atributo*	Indica el grado de riesgo que es capaz de asumir un avatar en sus acciones.
Personalidad.ConsultarPersonalidad ()	*Método*	Devuelve los rasgos de personalidad del avatar.
Personalidad.Crear (pers)	*Método*	Operación para crear un objeto de la clase Personalidad con los valores elegidos por el usuario.
Personalidad.Emocional	*Atributo*	Grado de emocionalidad del avatar. Varía de muy emocional a nada emocional.
Personalidad.Permisividad	*Atributo*	Indica el grado de permisividad de un avatar. Varía entre muy permisivo y nada permisivo.
Personalidad.SeleccionarPersonalidad (per)	*Método*	Rellena los rasgos de personalidad del avatar.
Personalidad.Simpatia	*Atributo*	Grado de simpatía del avatar. Varía entre muy simpático y nada simpático.
Personalidad.Sociabilidad	*Atributo*	Grado de sociabilidad del avatar. Varía entre muy sociable y nada sociable.

Personalidad.Timidez	*Atributo*	Grado de timidez del avatar. Varía entre muy tímido y nada tímido.
PiernaDe	*Clase*	Clase que representa la pierna derecha de los avatares.
PiernaIz	*Clase*	Clase que representa la pierna izquierda de los avatares.
Puerta	*Clase*	Punto de entrada y de salida del EV.
Puerta.ConsultarURL ()	*Método*	Devuelve la dirección a la que se va al atravesar la puerta.
Puerta.URL	*Atributo*	Lugar al que nos dirige la puerta al pasar por ella.
PuntoDeVista	*Clase*	Ofrece al usuario la capacidad de ver lo que sucede en el EV desde distintas posiciones.
PuntoDeVista.Crear (pv)	*Método*	Operación para crear un nuevo objeto de esta clase.
PuntoDeVista.Pintar ()	*Método*	Se encarga de pintar el EV cada vez que se produce un cambio, de manera que se muestre al usuario el nuevo estado en que se ha quedado el EV.
selecciona	*Asociación*	El Gestor de inicio está encargado de crear al Avatar.
Teclado	*Clase*	Dispositivo de conexión para controlar al avatar dentro del EV.
Teclado.Tecla_adelante	*Atributo*	Tecla para moverse hacia alante.
Teclado.Tecla_atrás	*Atributo*	Tecla para moverse hacia atrás.
Teclado.Tecla_correr	*Atributo*	Tecla para correr.
Teclado.Tecla_dcha	*Atributo*	Tecla para girar a la derecha.
Teclado.Tecla_izda	*Atributo*	Tecla para girar a la izquierda
tiene	*Asociación*	Un Avatar tiene asociados unos rasgos de Humor.
tiene	*Asociación*	Un Avatar tiene asociados unos rasgos de Personalidad.
tiene	*Asociación*	Un Avatar tiene una Memoria donde recordará, por ejemplo, los avatares a los que conoció en otras conexiones al EV.
tiene	*Asociación*	Un Avatar tiene un Cuerpo que lo representa en el EV.
tiene	*Asociación*	Un Avatar tiene un Cerebro para decidir qué acciones debe realizar y cómo las realiza.
Tronco	*Clase*	Clase que representa el tronco de los avatares.
usa	*Asociación*	Un Usuario usa un Dispositivo de conexión para controlar a su avatar dentro del EV.

utiliza	*Asociación*	Un Usuario utiliza un Punto de vista para ver el EV, de manera que su posición y su orientación le ofrecen distintas posibilidades para ver lo que sucede en el EV.

3. IMPLEMENTACIÓN

La utilización de WorldUp ha posibilitado que la implementación se haya podido hacer de una manera rápida y cómoda, y ha evitado tener que realizar a mano funcionalidades que ya se encontraban implementadas, como la comunicación a través de la red.

Sin embargo, el desconocimiento de algunas características de WorldUp a la hora de realizar el análisis y el diseño de este primer ciclo de desarrollo ocasionaron la aparición de algunos errores y características que no era necesario implementar, lo cual ya se encuentra resuelto en el diseño que se muestra de este primer ciclo de desarrollo.

También se encontraron algunos problemas a la hora de integrar la cámara de captura de movimiento con WorldUp, los cuales fueron subsanados sin necesidad de realizar ningún cambio en el diseño de la aplicación.

A continuación se presenta de manera un poco más detallada la forma en que se ha realizado la implementación, aunque sin entrar en los detalles de cómo se ha realizado la codificación.

3.1. CREACIÓN DE LOS MODELOS 3D

El primer paso para poder realizar cualquier labor con el EV es realizar la construcción de los modelos 3D que se han definido en la etapa de diseño, con los cuales se trabajará para que el EV vaya cobrando forma. Esta construcción se ha realizado en paralelo con el diseño del resto del primer ciclo, ya que ninguna de las decisiones que se tomen después del diseño de los avatares va a influir sobre ellos. En la Figura 38 ya hemos podido observar el aspecto que tenían los avatares que se han introducido para jugar al escondite inglés, por lo que nos queda por ver ahora el aspecto del escenario en el que se va a jugar. Este escenario se ha modelado utilizando Alias PowerAnimator, y el resultado obtenido es el que se observa en la Figura 70.

Figura 70: Habitación del Escondite Inglés

Para poder introducir los modelos en el EV, ha sido necesario crear dos nuevos tipos de objetos que no existen en WorldUp, según se presenta en la Figura 71.

Figura 71: Nuevos Objetos en WorldUp

Los grupos *Avatar* y *Habitación* nos servirán para tratar a los avatares y al escenario respectivamente. El grupo *Partida* servirá para guardar los datos relativos a la partida en curso. A pesar de representar la partida como si fuese un objeto físico del EV, no lo va a ser, pero esta es la única manera en que WorldUp permite introducir elementos que serán necesarios de cara a la ejecución del sistema. Por debajo del grupo *Avatar* se irán creando los avatares y sus respectivas partes del cuerpo cuando se carguen éstos dentro del EV.

De la misma manera que se ha hecho esto, se han creado nuevos grupos para representar la *Memoria*, el *Humor*, la *Personalidad* y el *Cerebro* de los avatares.

Por lo que respecta a las luces u los puntos de vista, son algo que ya está incluido en otra rama de la jerarquía de objetos predefinidos de WorldUp, por lo que, aunque no respetan el diseño que se ha realizado previamente, son suficientes como para cumplir la función que deben cumplir.

Por último, los objetos relativos a los usuarios y las conexiones son también propios de WorldUp, por lo que no hay que preocuparse de su creación y su gestión. Igual sucede con el *Chat*, el cual es una aplicación adicional que viene incluida con WorldUp y World2World.

3.2. CREACIÓN DE MOVIMIENTOS

Una vez que ya tenemos los modelos introducidos, es hora de ponerse a trabajar con ellos. Para ello, lo que se ha hecho es diseñar una serie de movimientos con el editor de *paths* que incluye WorldUp, según se puede ver en la Figura 72.

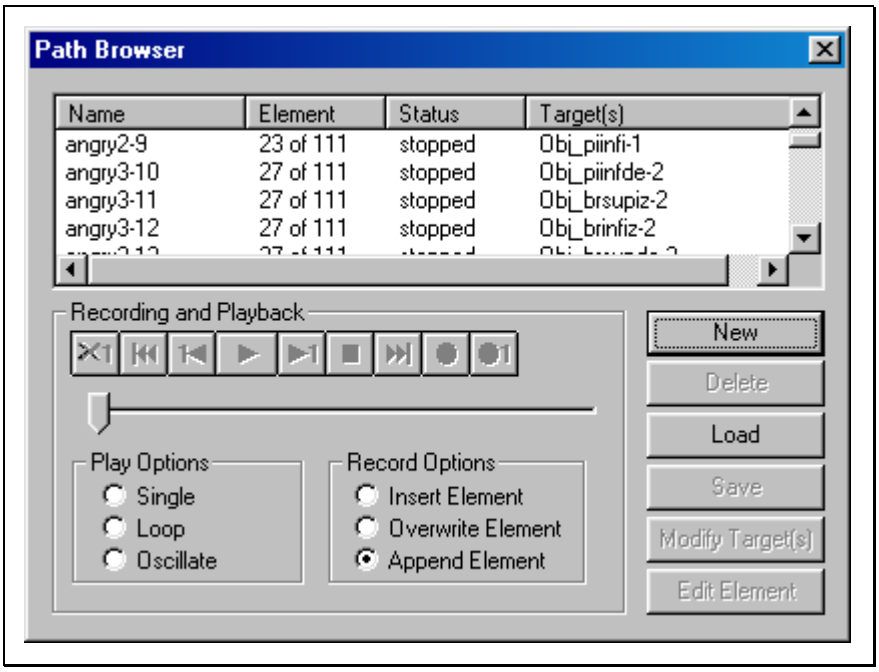

Figura 72: Editor de *paths*

De esta manera se ha creado toda la batería de movimientos descritos en la fase de diseño que pueden realizar los avatares dentro del EV. Hay que destacar que cada path está asociado a un objeto concreto del EV, y además almacena coordenadas absolutas del mundo, razón por la cual hay que diseñar cada movimiento específicamente para cada avatar, lo cual es un inconveniente bastante grande, pues no permite generalizar las acciones, lo cual sería la situación deseable.

3.3. INTERACCIÓN CON LA CÁMARA

La interacción con la cámara ha sido uno de los elementos más complicados que ha habido que tener en cuenta a la hora de realizar la implementación, ya que WorldUp no incorpora ningún mecanismo para interactuar con un elemento de este tipo. Por esta razón, ha sido necesario programar una librería dinámica en C++ para permitir la comunicación con este dispositivo de entrada, pues WorldUp posee el mecanismo necesario para poder utilizar este tipo de librerías.

Por otro lado, la especial geometría de la lente utilizada por la cámara ha ocasionado que el calibrado de la cámara y la transformación de medidas del mundo real a medidas del EV se haya tenido que realizar de manera empírica. El procedimiento seguido es el que se detalla a continuación.

- En primer lugar, es necesario hallar el centro de masas del usuario dentro de la habitación, ya que éste va a ser quien defina la posición del usuario dentro de la habitación.
- La habitación de juego se ha dividido en secciones de 60 cm, y para cada uno de los límites de esas secciones se ha obtenido la correspondiente posición del centro de masas en la imagen, como se presenta en la siguiente tabla:

X **Pixel**	**Y** **Centímetros**
21	120
37	180
59	240
122	300
193	360
293	420
489	480
621	540
706	600

- Una vez obtenidos los valores de la tabla, y en vista de que no siguen una relación lineal, se ha optado por construir una función a trozos para obtener la correspondencia de los valores intermedios, y cuya forma se puede ver en la Figura 73. La construcción de esta función se ha realizado siguiendo el método que se explica a continuación:

1. Para cada X_{dato} vamos a obtener un valor para ***u,*** usando la siguiente fórmula:

$$u = (X_{dato} - X_{inferior}) / (X_{inferior} - X_{superior})$$

donde $X_{inferior}$ es el valor de X, en la tabla, más cercano por debajo y $X_{superior}$ es el siguiente valor X en la tabla. La Y correspondiente a $X_{inferior}$ es $Y_{inferior}$, y para $X_{superior}$ es $Y_{superior}$.

2. Ahora calculamos el valor de Y, aplicando la fórmula:

$$Y = Y_{inferior} + 60*u$$

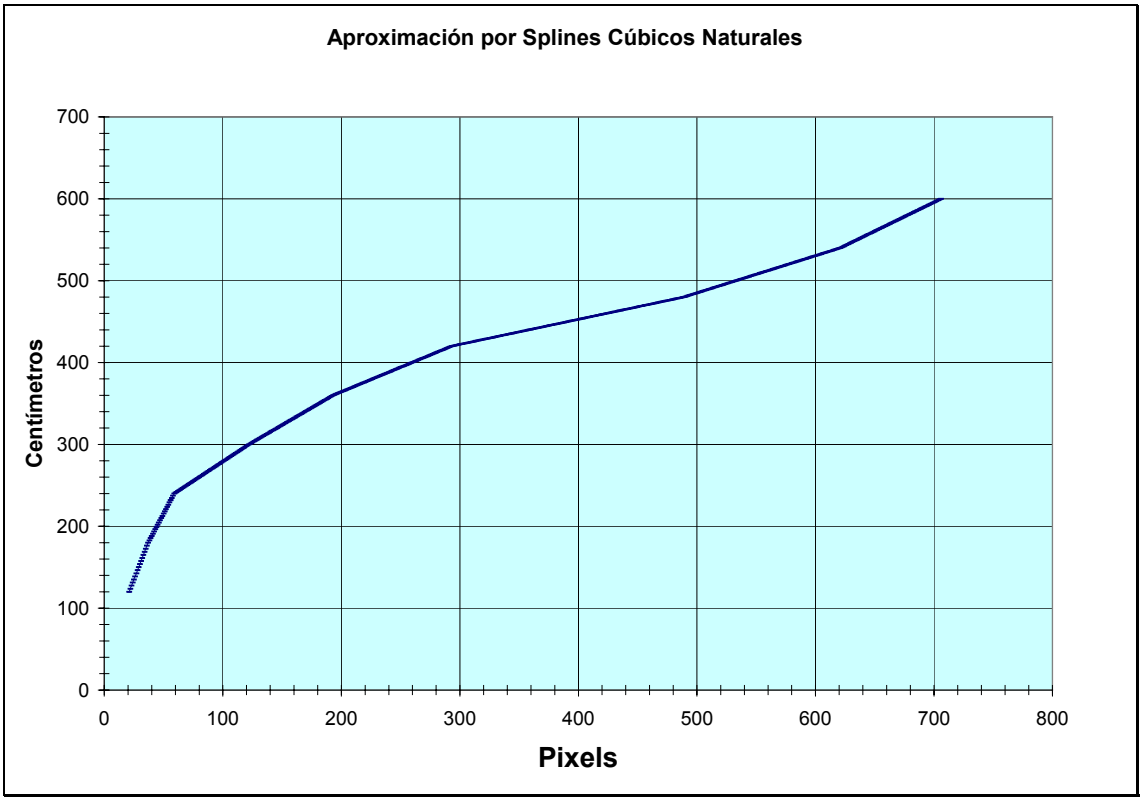

Figura 73: Gráfica de la función de conversión

- Como se ve, de la posición del centro de masas en la imagen se obtiene la posición del usuario en la habitación de juego. Puesto que conocemos la dimensión de ésta y de la habitación del EV, el pasar la posición de una a otra se realiza de forma inmediata, sin más que realizar una proporción entre ambas.

3.4. ADICIÓN DE COMPORTAMIENTO A LOS OBJETOS

Para añadir comportamiento a los objetos, lo que se ha hecho es asociarles scripts que se encarguen de la comprobación y la realización de ciertas labores, como es la detección de los distintos eventos dentro del EV o la respuesta a pulsaciones de los botones del interfaz.

3.4.1. Comportamiento de los Avatares

El comportamiento de los avatares se ha implementado con la utilización de 2 *scripts*:

- ♣ controlScript: se encarga de la realización de acciones ordenadas por el usuario.
- ♣ moveScript: está activo sólo cuando el avatar no se la liga. Está a cargo de controlar cómo se realiza el movimiento del avatar por el EV, y es también donde se realiza la comunicación con el dispositivo de conexión.

3.4.2. Comportamiento de la Partida

La partida es otro de los objetos que tendrán gran importancia dentro del EV creado con WorldUp. Por motivos de eficiencia, en este mismo objeto se realizan las acciones relacionadas con el gestor de inicio.

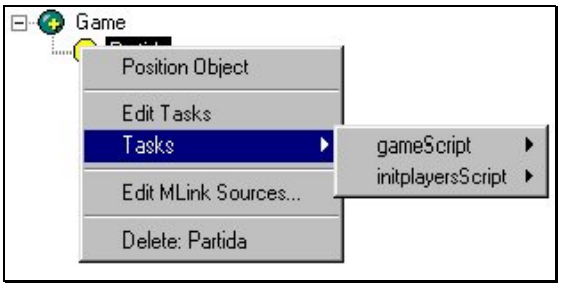

Figura 74: Comportamiento del objeto Partida

- ♣ initplayersScript: se encarga de todas las labores de inicialización al realizar la conexión al EV.
- ♣ gameScript: se encarga de realizar todas las labores que corresponden a un objeto de la clase partida.

SEGUNDO CICLO DE DESARROLLO

1. ANÁLISIS

Después de haber llevado a cabo el primer ciclo de desarrollo, en el cual se han tratado todos los aspectos relacionados con la interacción del usuario con el sistema, es necesario realizar un segundo ciclo en el cual se amplía lo que se ha hecho hasta ahora para permitir que un usuario ceda parte del control de su avatar a la máquina, de manera que pueda estar menos pendiente de controlar a su avatar y pueda centrarse más en lo que sucede dentro del EV.

Como ya se ha comentado anteriormente, esto no es posible hacerlo con una metodología dirigida por casos de uso, como es la de Larman [Larman99], razón por la cual se van a utilizar algunas de las herramientas definidas en [Sánchez99]. Sin embargo, puesto que lo que se pretendía en un principio era seguir el método definido por Larman, lo que se va a intentar hacer en este segundo ciclo de desarrollo es utilizar elementos externos a UML y adaptarlos lo más posible al proceso definido por Larman.

En base a esto, se hará una analogía entre casos de uso y conceptos de uso, para definir una visión global de las operaciones. A partir de los conceptos de uso se realizarán una serie de escenarios, de manera similar a como se hacían los diagramas de secuencia del sistema a partir de los casos de uso. También se realizarán, para dotar de más claridad al análisis, diagramas de estados sobre las acciones representadas en los escenarios, pues aunque no aparecen entre los elementos que recomienda Larman para esta fase [Larman99], son elementos opcionales que plantea para la misma.

Por último, siguiendo otra vez con el método de Larman, se extenderá el modelo conceptual obtenido en el primer ciclo del presente desarrollo y, para concluir con el análisis de este segundo ciclo, se especificarán los contratos de las nuevas operaciones, tras lo cual se iniciará la fase de diseño.

1.1. ESCENARIOS

Como mecanismo utilizado para sustituir a los diagramas de secuencia del sistema, la diferencia básica con éstos estribará en la ausencia de un actor externo al sistema que sea quien inicia las operaciones, sino que en cada operación será una parte del propio sistema la que tenga que iniciar la acción y que, con toda probabilidad, será un elemento que posteriormente hará su aparición en el modelo conceptual. Puesto que

en principio lo que se va a automatizar va a ser el comportamiento del avatar, el elemento que se va a utilizar como iniciador de las acciones será el cerebro que tiene asociado cada avatar, aunque al final no tendrá por qué coincidir con la clase *Cerebro* que aparece en el diagrama de clases de diseño del ciclo anterior.

Por otro lado, hay que hacer notar que la diferencia básica entre los escenarios y los diagramas de secuencia del sistema es que con los escenarios se está representando una operación interna del sistema, razón por la cual los mensajes que aparecen en ellos no van a corresponder a una operación concreta del sistema como sucedía con los diagramas de secuencia del sistema, cuyos mensajes se convertían en la operación iniciadora de una acción representada en un diagrama de interacción. En este caso, lo que va a suceder es que el mensaje enviado acabará siendo descompuesto en otra serie de operaciones en los diagramas de interacción, pero no dará lugar a un mensaje que inicie una acción.

Esto es así porque las acciones no van realizarse por iniciativa de un usuario, sino de un objeto, que además será el objeto que se encargue de realizar la acción. Si se representase aquí toda la secuencia de la acción, entonces carecería de sentido utilizar más tarde los diagramas de interacción. Por esta razón, se ha decidido representar en los escenarios la acción que se debe realizar, y posteriormente, en los diagramas de interacción, se descompondrá la misma para determinar la forma exacta en que ésta se realiza.

1.1.1. Concepto de Uso *Detectar avatar conocido*

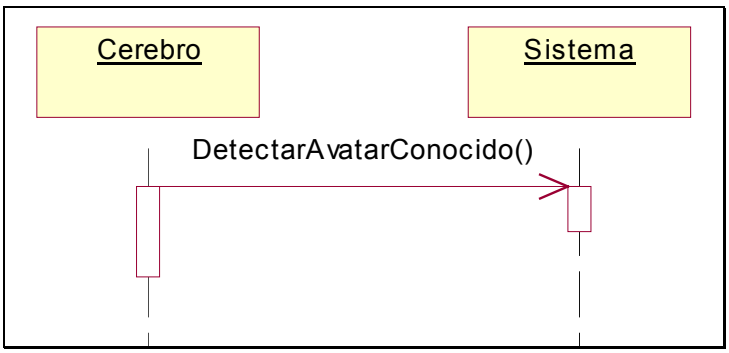

Figura 75: Detectar avatar conocido

1.1.2. Concepto de Uso *Saludar a un avatar*

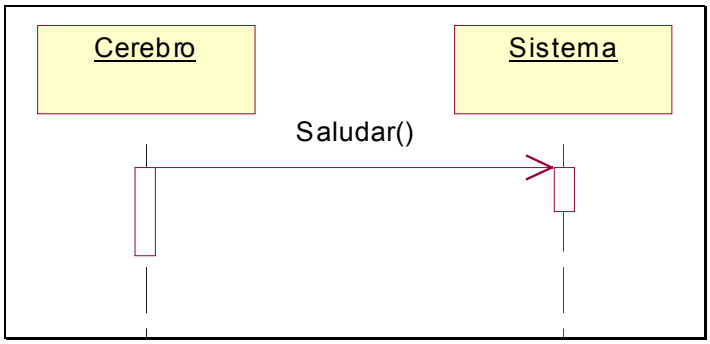

Figura 76: Saludar

1.1.3. Concepto de Uso *Detectar saludo*

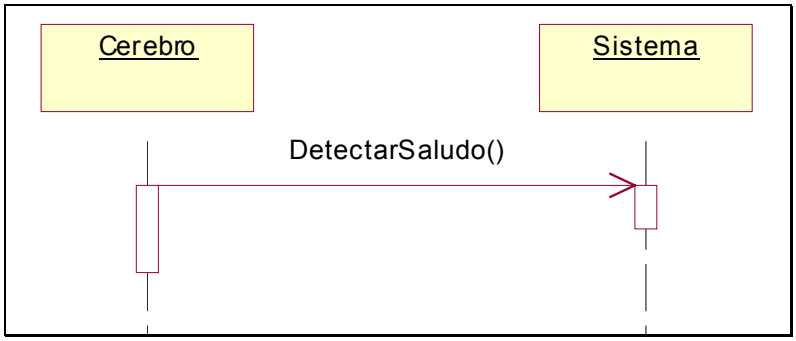

Figura 77: Detectar saludo

1.1.4. Concepto de Uso *Hacer reír*

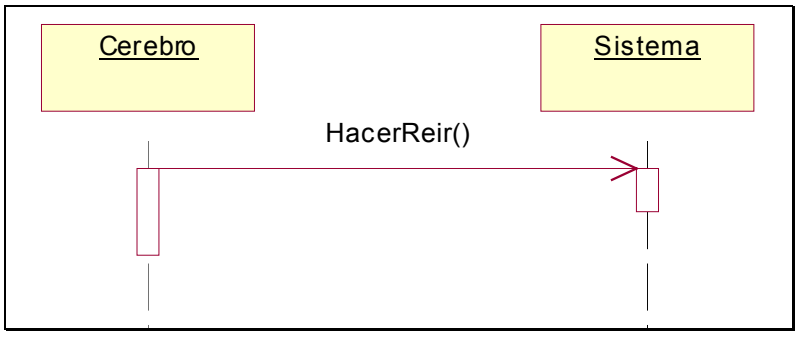

Figura 78: Hacer reír

1.1.5. Concepto de Uso *Detectar que me hacen reír*

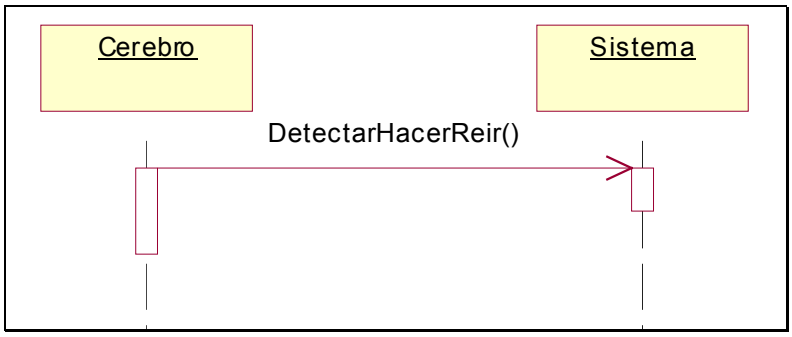

Figura 79: Detectar que me hacen reir

1.1.6. Concepto de Uso *Reír*

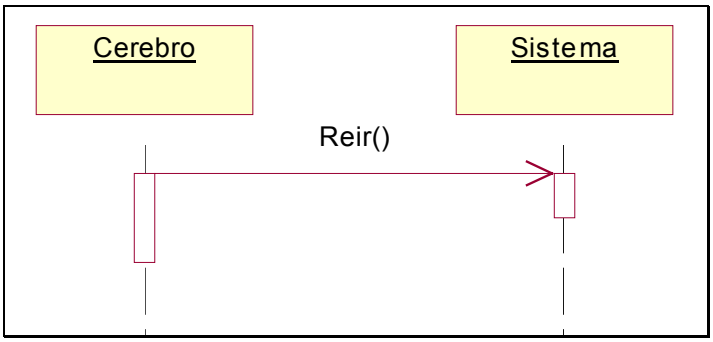

Figura 80: Reir

1.1.7. Concepto de Uso *Saltar de Alegría*

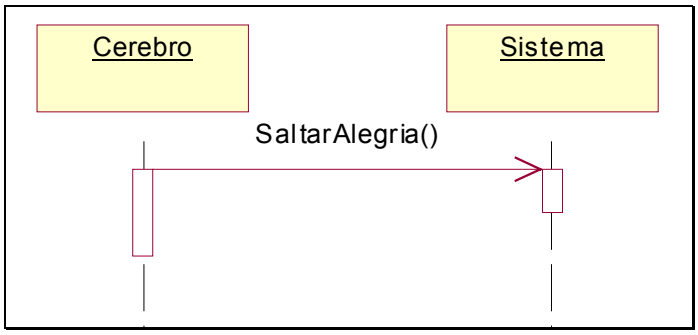

Figura 81: Saltar de Alegría

1.1.8. Concepto de Uso *Bailar*

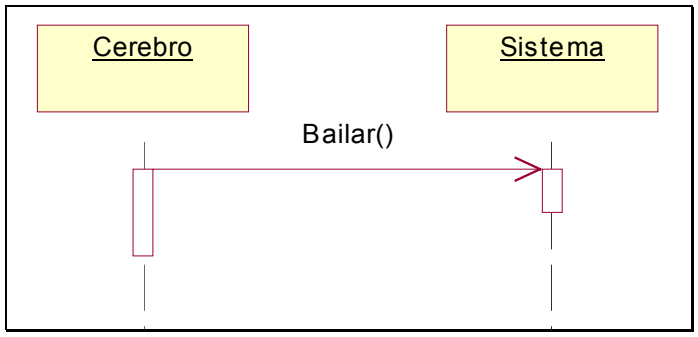

Figura 82: Bailar

1.1.9. Concepto de Uso *Ponerse triste*

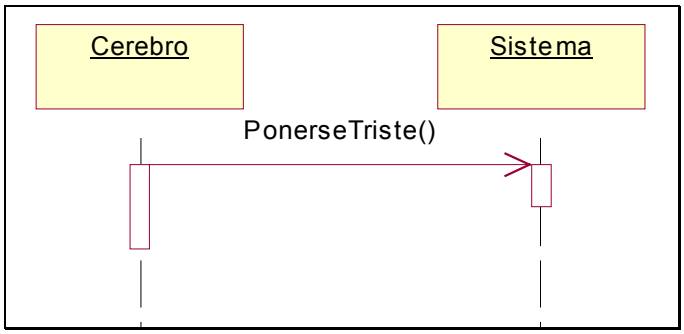

Figura 83: Ponerse triste

1.1.10. Concepto de Uso *Detectar avatar triste*

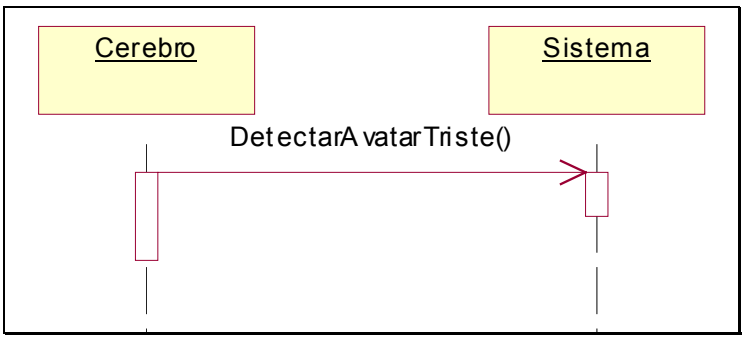

Figura 84: Detectar avatar triste

1.1.11. Concepto de Uso *Llorar*

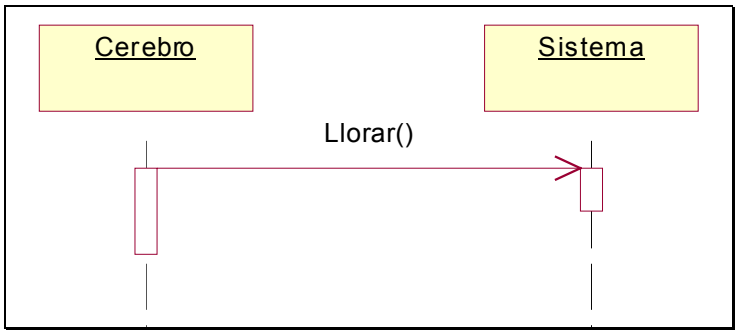

Figura 85: Llorar

1.1.12. Concepto de Uso *Enfadarse*

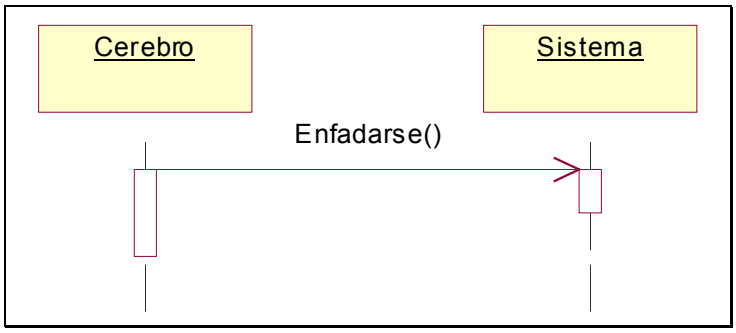

Figura 86: Enfadarse

1.1.13. Concepto de Uso *Detectar avatar enfadado*

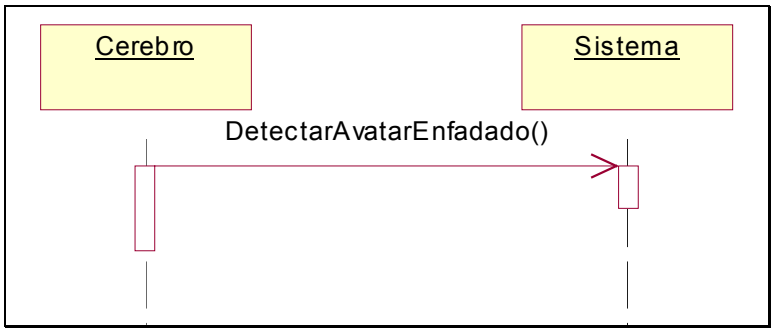

Figura 87: Detectar avatar enfadado

1.1.14. Concepto de Uso *Patalear*

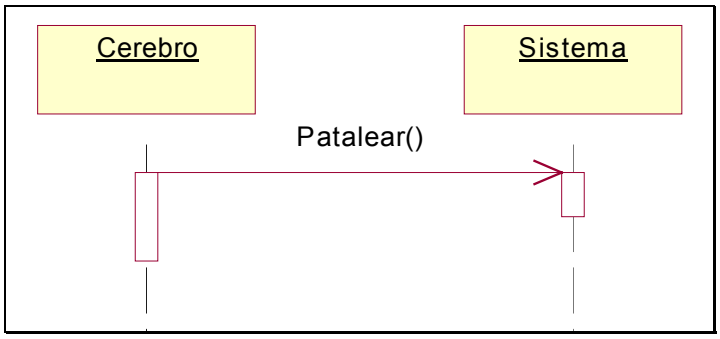

Figura 88: Patalear

1.1.15. Concepto de Uso *Agredir*

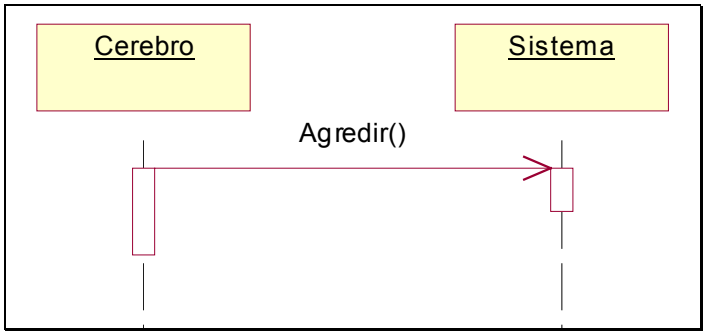

Figura 89: Agredir

1.1.16. Concepto de Uso *Detectar agresión*

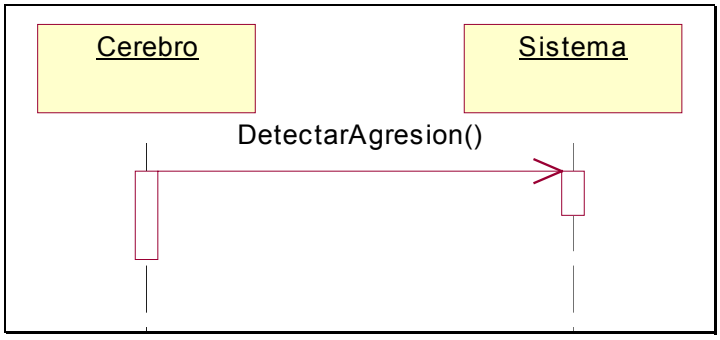

Figura 90: Detectar agresión

1.1.17. Concepto de Uso *Aburrirse*

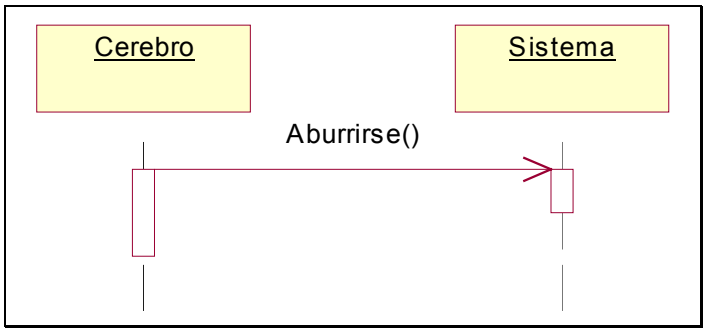

Figura 91: Aburrirse

1.1.18. Concepto de Uso *Detectar avatar aburrido*

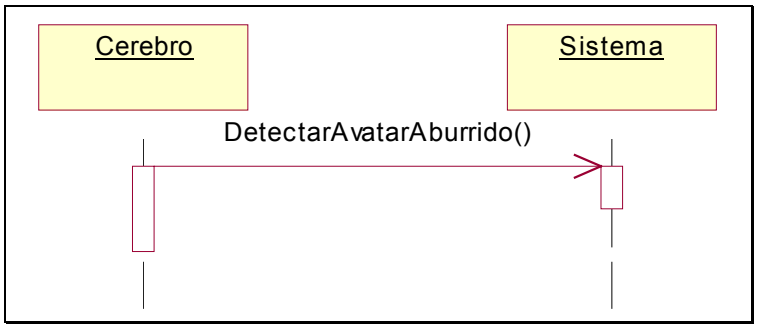

Figura 92: Detectar avatar aburrido

1.1.19. Concepto de Uso *Detectar gente alrededor*

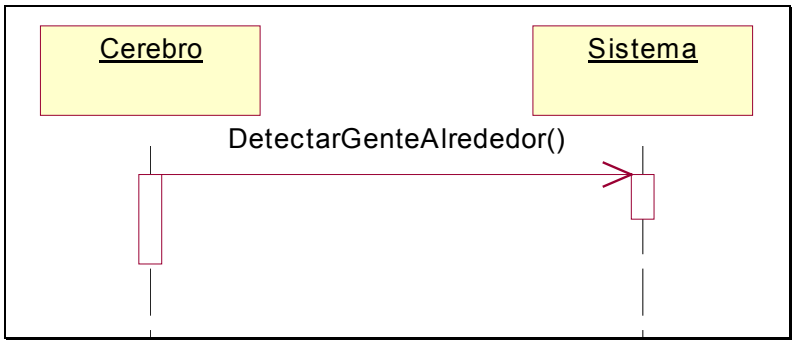

Figura 93: Detectar Gente Alrededor

1.1.20. Concepto de Uso *Detectar avatar lejos de la pared*

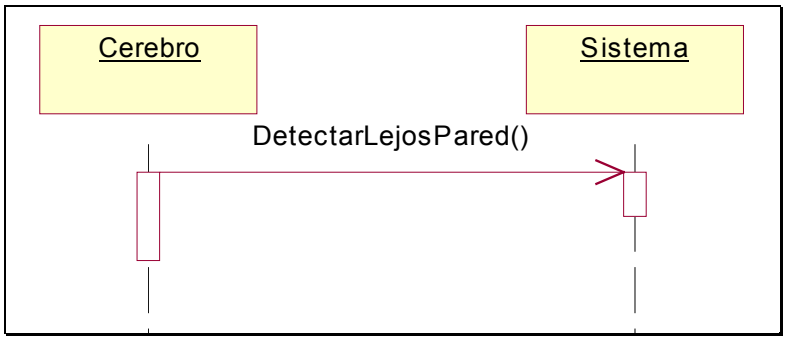

Figura 94: Detectar avatar lejos de la pared

1.1.21. Concepto de Uso *Detectar avatar cerca de la pared*

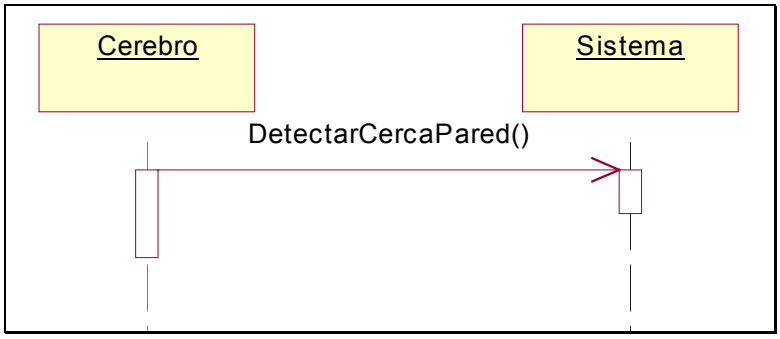

Figura 95: Detectar avatar cerca de la pared

1.1.22. Concepto de Uso *Detectar otros avatares cerca de la pared*

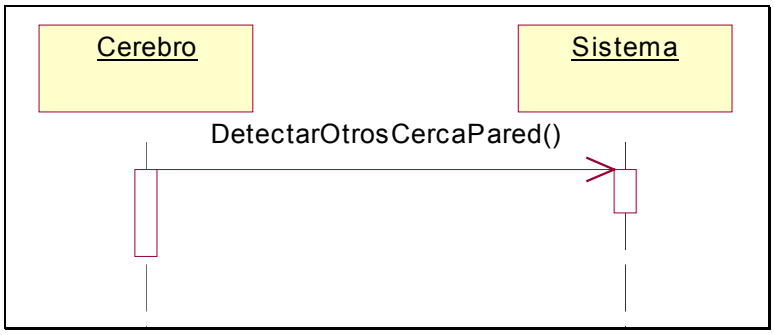

Figura 96: Detectar otros avatares cerca de la pared

1.1.23. Concepto de Uso *Detectar jugadores moviéndose*

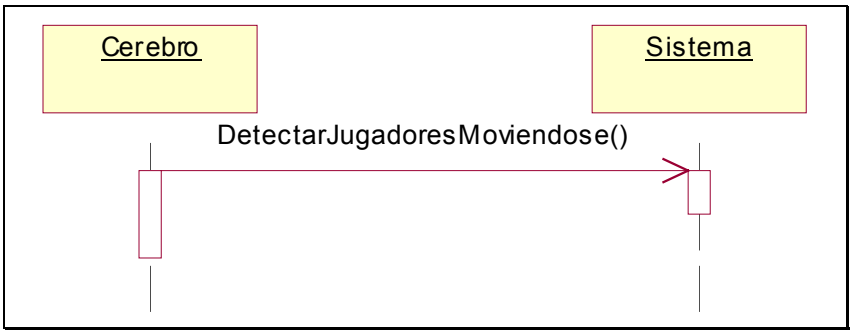

Figura 97: Detectar jugadores moviéndose

1.1.24. Concepto de Uso *Detectar avatar visto moviéndose*

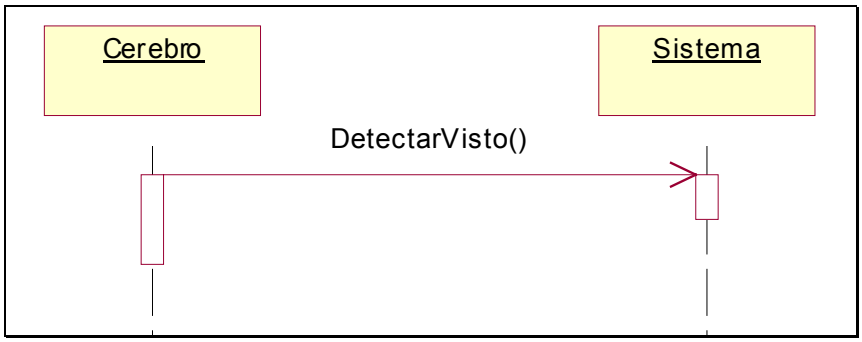

Figura 98: Detectar avatar visto moviéndose

1.1.25. Concepto de Uso *Volver a la línea de partida*

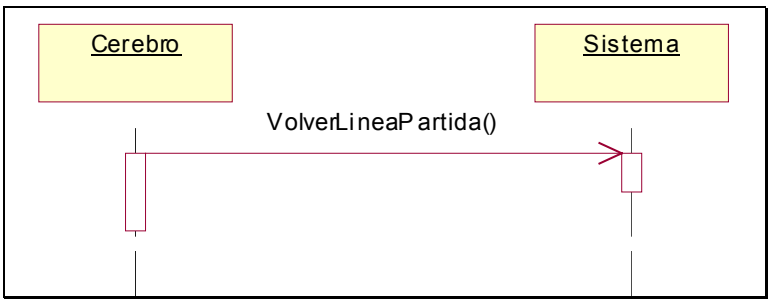

Figura 99: Volver a la línea de partida

1.1.26. Concepto de Uso *Detectar demasiadas veces visto*

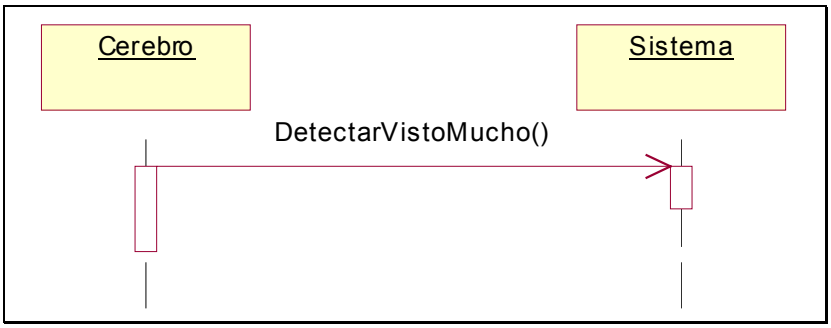

Figura 100: Detectar demasiadas veces visto

1.1.27. Concepto de Uso *Detectar duración de la partida*

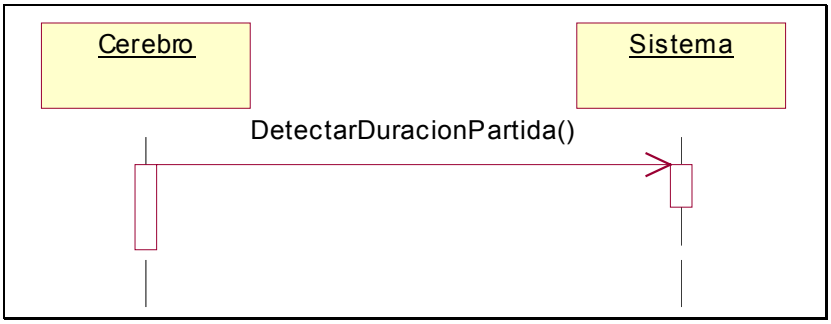

Figura 101: Detectar duración de la partida

1.1.28. Concepto de Uso *Detectar avatar gana el juego*

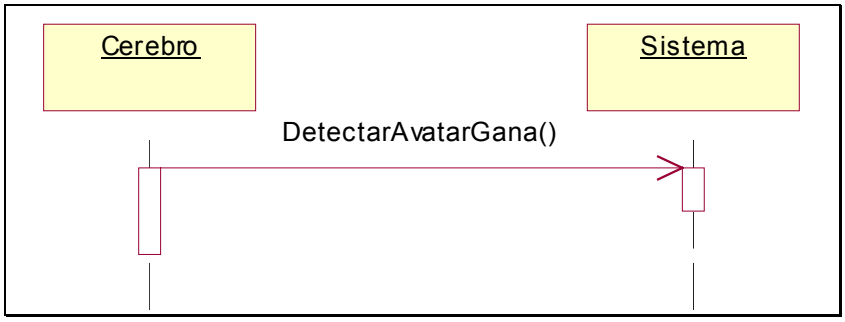

Figura 102: Detectar avatar gana el juego

1.1.29. Concepto de Uso *Detectar avatar pierde el juego*

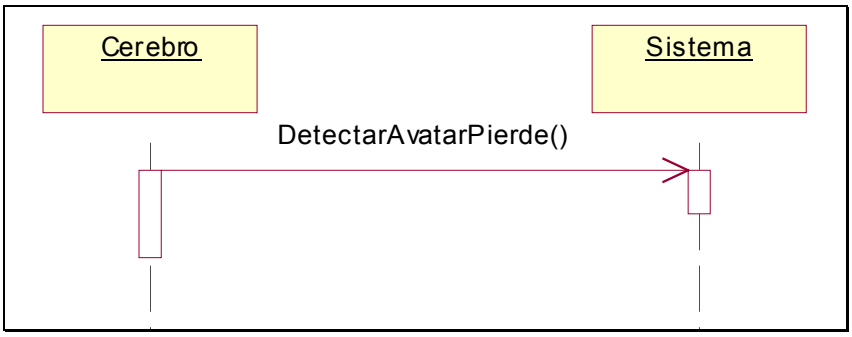

Figura 103: Detectar avatar pierde el juego

1.1.30. Concepto de Uso *Volverse hacia la pared para contar*

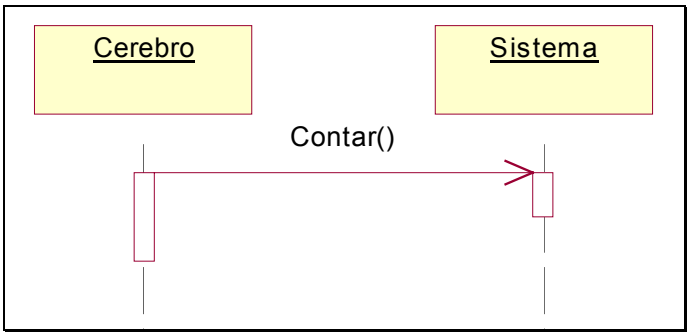

Figura 104: Volverse hacia la pared para contar

1.1.31. Concepto de Uso *Detectar que el que se la liga está contando*

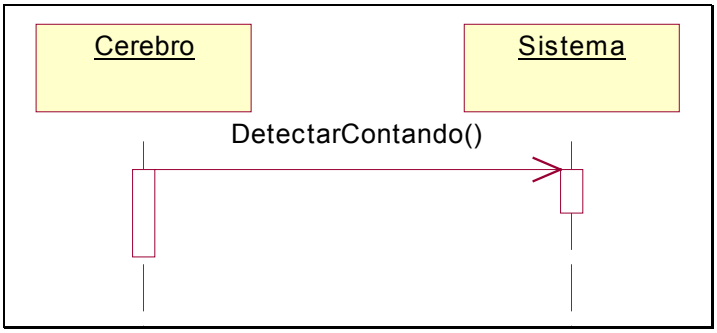

Figura 105: Detectar que el que se la liga está contando

1.1.32. Concepto de Uso *Volverse hacia los jugadores*

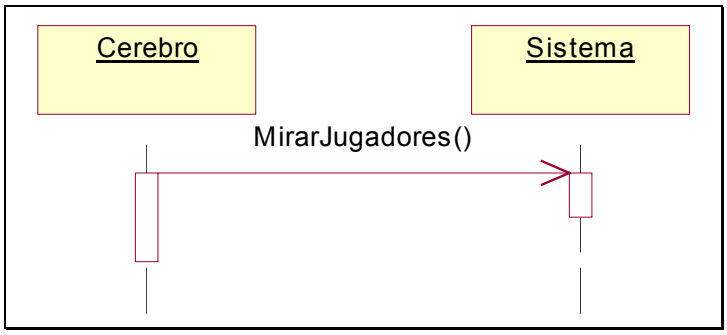

Figura 106: Volverse hacia los jugadores

1.1.33. Concepto de Uso *Decidir quién se la liga*

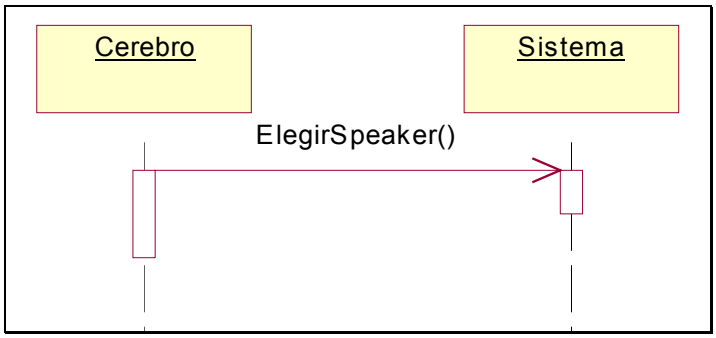

Figura 107: Decidir quién se la liga

1.1.34. Concepto de Uso *Detectar obstáculo*

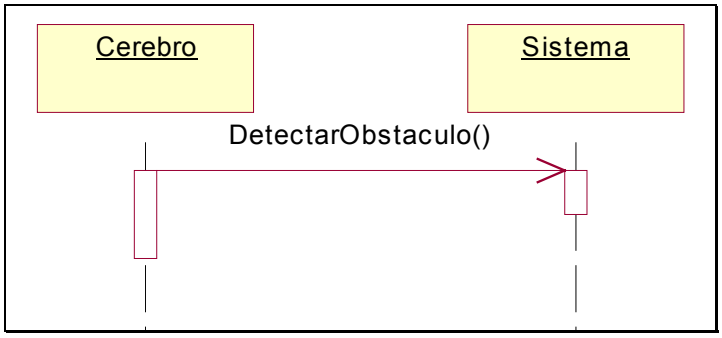

Figura 108: Detectar obstáculo

1.2. DIAGRAMAS DE ESTADOS

Este es uno de los elementos proporcionados por UML de los cuales Larman no hace un uso directo dentro de su método de desarrollo [Larman99]; pese a todo, sí menciona la posibilidad de utilizarlos en la etapa de análisis para modelar la evolución que puede tener una parte del sistema o éste en su totalidad a partir de una serie de acciones descritas en los casos de uso.

En el presente desarrollo, se van a utilizar estos diagramas para describir el comportamiento que va a tener un avatar dependiendo de su estado de humor y de su personalidad. La información se extraerá de la definición de los conceptos de uso y del modelo de personalidad descrito en [Imbert98].

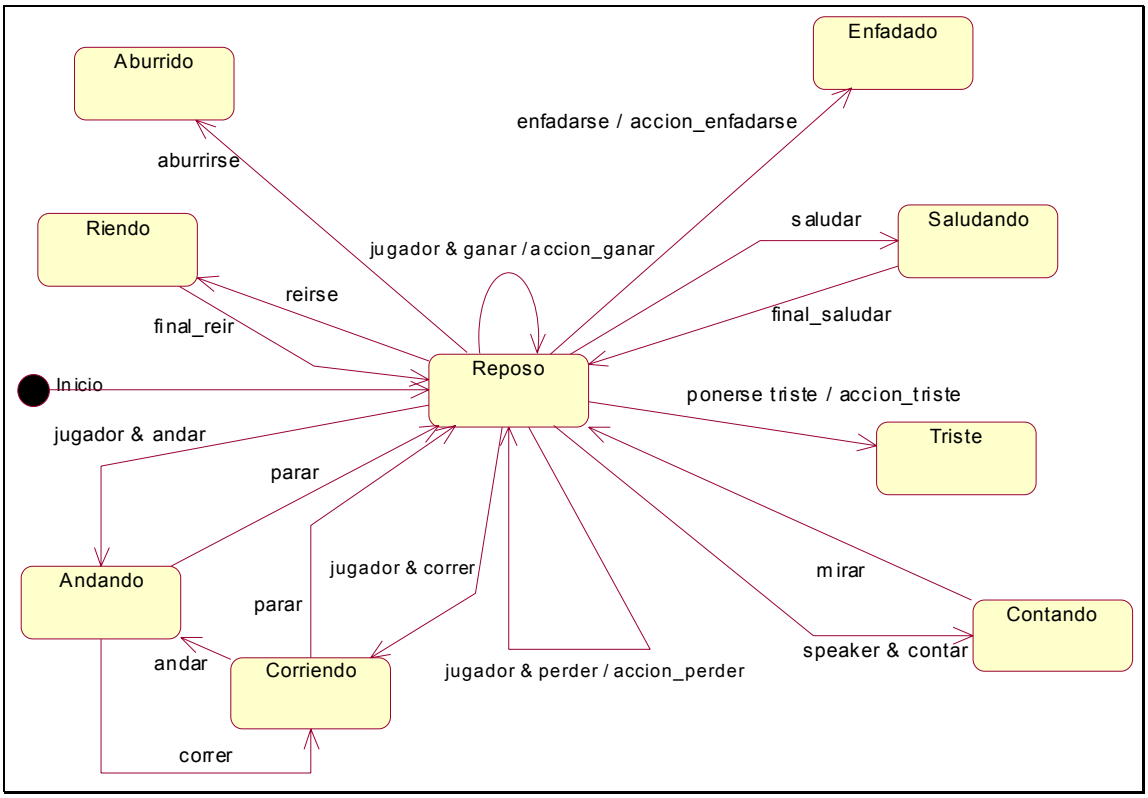

Figura 109: Diagrama de estados

1.3. MODELO CONCEPTUAL

Como ampliación al modelo conceptual presentado en el primer ciclo de desarrollo, se ha considerado que deben hacer su aparición los mecanismos reales de los cuales dispondrá un avatar para realizar ciertas labores de manera automática, como son los sentidos que le permiten ver y oír lo que sucede en el EV. También dispondrá de un cerebro que será el que, como se ha visto en los escenarios, coordine las acciones que tiene que realizar el avatar.

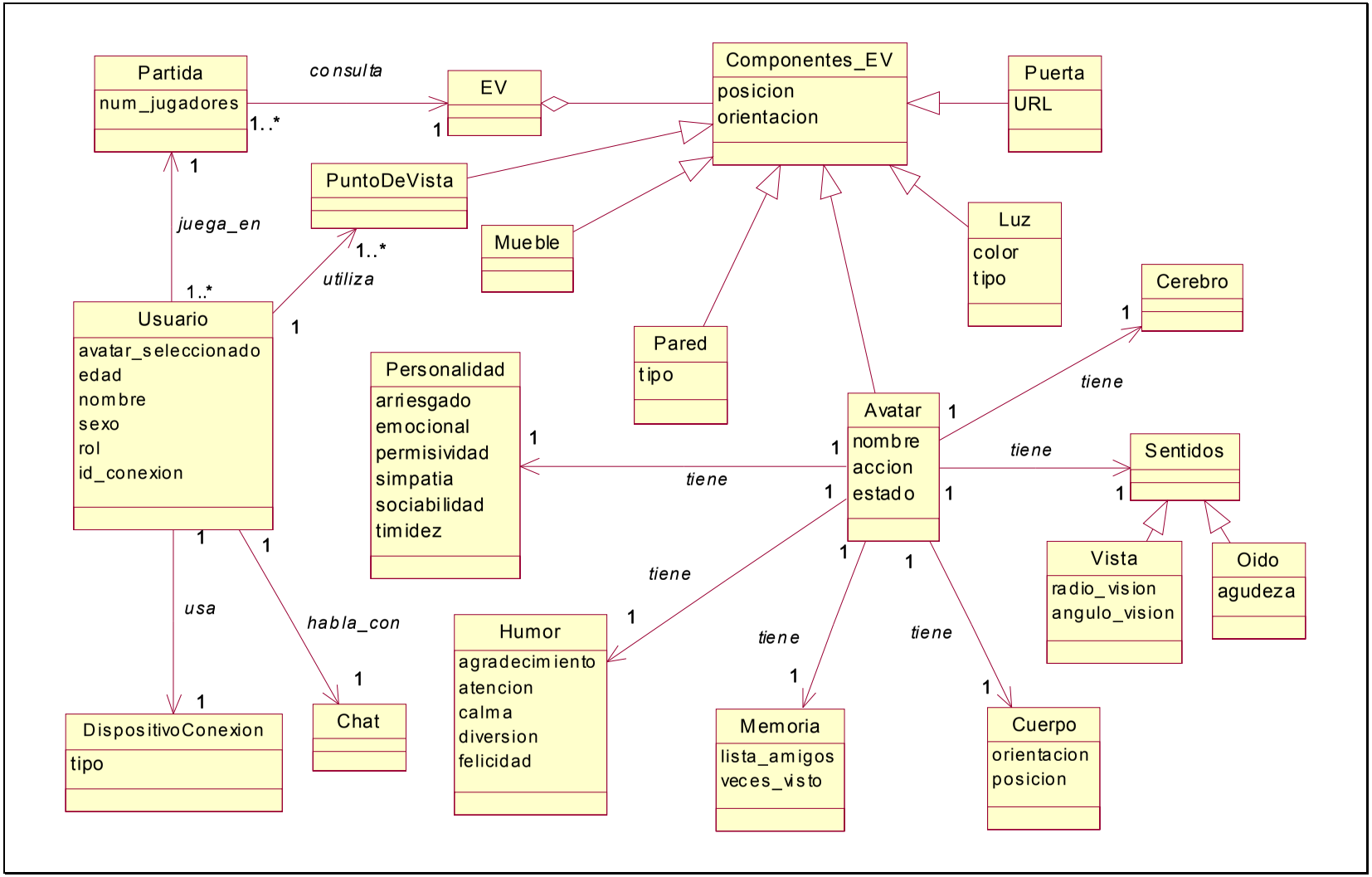

Figura 110: Ampliación del Modelo Conceptual

1.4. CONTRATOS DE OPERACIONES

De forma análoga a como se hacía para el caso de los diagramas de secuencia del sistema, para cada una de las operaciones aparecidas en los escenarios se va a describir un contrato de operación que será desarrollado en la etapa de diseño para incluir en el desarrollo las nuevas funcionalidades que se requieren para los avatares, esto es, la automatización de ciertas tareas que pueden llevar a cabo.

Nombre	DetectarAvatarConocido ()
Responsabilidades	Comprobar si hay conectado algún avatar al que se conozca de otras conexiones.
Tipo	Sistema
Referencias Cruzadas	Concepto de uso *Detectar avatar conocido*.
Excepciones	
Precondiciones	↳ El usuario debe estar conectado al EV.
Postcondiciones	

Nombre	Saludar ()
Responsabilidades	Saludar a los avatares que se encuentren conectados y aparezcan dentro de la lista de amigos.
Tipo	Sistema
Referencias Cruzadas	Concepto de uso *Saludar a un avatar*.
Excepciones	
Precondiciones	↳ Se deben haber detectado avatares conocidos. ↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se ha cambiado el valor de la acción del avatar.

Nombre	DetectarSaludo ()
Responsabilidades	Detectar si otro avatar nos está saludando y responder a su saludo si se considera necesario.
Tipo	Sistema
Referencias Cruzadas	Concepto de uso *Detectar saludo*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se ha cambiado el valor de la acción del avatar.

Nombre	HacerReir ()
Responsabilidades	Intentar provocar la risa a otro avatar.
Tipo	Sistema
Referencias Cruzadas	Concepto de uso *Hacer reír*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se ha cambiado el valor de la acción del avatar.

Nombre	DetectarHacerReir ()
Responsabilidades	Detectar si otro avatar me está intentando hacer reír y reírse si lo consigue.
Tipo	Sistema
Referencias Cruzadas	Concepto de uso *Detectar que me hacen reír*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se ha cambiado el valor de la acción del avatar.

Nombre	Reir ()
Responsabilidades	Hacer que mi avatar se ría.
Tipo	Sistema
Referencias Cruzadas	Concepto de uso *Reír*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se ha cambiado el valor de la acción del avatar.

Nombre	SaltarAlegria ()
Responsabilidades	Hacer que el avatar realice el gesto de saltar de alegría.
Tipo	Sistema
Referencias Cruzadas	Concepto de uso *Saltar de Alegría*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se ha cambiado el valor de la acción del avatar.

Nombre	Bailar ()
Responsabilidades	Hacer que el avatar realice la acción de bailar.
Tipo	Sistema
Referencias Cruzadas	Concepto de uso *Bailar.*
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se ha cambiado el valor de la acción del avatar.

Nombre	PonerseTriste ()
Responsabilidades	Hacer que el avatar muestre que está triste.
Tipo	Sistema
Referencias Cruzadas	Concepto de uso *Ponerse triste*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se han cambiado los valores del humor del avatar.

Nombre	DetectarAvatarTriste ()
Responsabilidades	Detectar si hay algún avatar triste en el EV y obrar en consecuencia, de acuerdo con el humor y la personalidad.
Tipo	Sistema
Referencias Cruzadas	Concepto de uso *Detectar avatar triste*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se ha cambiado el valor de la acción del avatar.

Nombre	Llorar ()
Responsabilidades	Hacer que el avatar realice la acción de llorar.
Tipo	Sistema
Referencias Cruzadas	Concepto de uso *Llorar*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se ha cambiado el valor de la acción del avatar. ↳ Se han cambiado los valores del humor del avatar.

Nombre	Enfadarse ()
Responsabilidades	Mostrar el estado de enfado de un avatar.
Tipo	Sistema
Referencias Cruzadas	Concepto de uso *Enfadarse*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se ha cambiado el valor de la acción del avatar. ↳ Se han cambiado los valores del humor del avatar.

Nombre	DetectarAvatarEnfadado ()
Responsabilidades	Detectar si hay algún avatar enfadado dentro del EV y obrar en consecuencia, de acuerdo con el humor y la personalidad.
Tipo	Sistema
Referencias Cruzadas	Concepto de uso *Detectar avatar enfadado*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se ha cambiado el valor de la acción del avatar.

Nombre	Patalear ()
Responsabilidades	Hacer que el avatar realice la acción de tirarse al suelo y patalear.
Tipo	Sistema
Referencias Cruzadas	Concepto de uso *Patalear*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se ha cambiado el valor de la acción del avatar. ↳ Se han cambiado los valores del humor del avatar.

Nombre	Agredir ()
Responsabilidades	Hacer que el avatar realice la acción de agredir al otro avatar.
Tipo	Sistema
Referencias Cruzadas	Concepto de uso *Agredir*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se ha cambiado el valor de la acción del avatar. ↳ Se han cambiado los valores del humor del avatar.

Nombre	DetectarAgresion ()
Responsabilidades	Detectar que otro avatar me está agrediendo.
Tipo	Sistema
Referencias Cruzadas	Concepto de uso *Detectar agresión*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se ha cambiado el valor de la acción del avatar.

Nombre	Aburrirse ()
Responsabilidades	Aumentar el grado de aburrimiento del avatar y mostrarlo externamente.
Tipo	Sistema
Referencias Cruzadas	Concepto de uso *Aburrirse*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se ha cambiado el valor de la acción del avatar. ↳ Se han cambiado los valores del humor del avatar.

Nombre	DetectarAvatarAburrido ()
Responsabilidades	Detectar si hay algún avatar en el EV que se esté aburriendo y obrar de acuerdo con el humor y la personalidad.
Tipo	Sistema
Referencias Cruzadas	Concepto de uso *Detectar avatar aburrido*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se ha cambiado el valor de la acción del avatar.

Nombre	DetectarGenteAlrededor ()
Responsabilidades	Detectar si hay gente cerca y reaccionar de acuerdo a la personalidad y al humor.
Tipo	Sistema
Referencias Cruzadas	Concepto de uso *Detectar gente alrededor*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se han cambiado los valores del humor del avatar.

Nombre	DetectarLejosPared ()
Responsabilidades	Ver si el avatar está lejos de la pared y reaccionar de acuerdo con la personalidad y el humor.
Tipo	Sistema
Referencias Cruzadas	Concepto de uso *Detectar avatar lejos de la pared*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se han cambiado los valores del humor del avatar.

Nombre	DetectarCercaPared ()
Responsabilidades	Detectar si el avatar está cerca de la pared y reaccionar de acuerdo con los valores de la personalidad y del humor.
Tipo	Sistema
Referencias Cruzadas	Concepto de uso *Detectar avatar cerca de la pared*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se han cambiado los valores del humor del avatar.

Nombre	DetectarOtrosCercaPared ()
Responsabilidades	Detectar si hay otros avatares más cerca de la pared que yo y obrar de acuerdo con los valores del humor y de la personalidad.
Tipo	Sistema
Referencias Cruzadas	Concepto de uso *Detectar otros avatares cerca de la pared*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se han cambiado los valores del humor del avatar.

Nombre	DetectarJugadoresMoviendose ()
Responsabilidades	Ordenar a los jugadores que se estén moviendo que vuelvan a la línea de comienzo.
Tipo	Sistema
Referencias Cruzadas	Concepto de uso *Detectar jugadores moviéndose*.
Excepciones	
Precondiciones	↳ Que el avatar se la ligue. ↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	

Nombre	DetectarVisto ()
Responsabilidades	Detectar que el avatar que se la liga me ha visto moviéndome.
Tipo	Sistema
Referencias Cruzadas	Concepto de uso *Detectar avatar visto moviéndose*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se han cambiado los valores del humor del avatar. ↳ Se incrementa el número de veces que me han visto moviéndome que se almacena en la memoria.

Nombre	VolverLineaPartida ()
Responsabilidades	Hacer que el avatar vuelva a la línea de comienzo.
Tipo	Sistema
Referencias Cruzadas	Concepto de uso *Volver a la línea de partida*.
Excepciones	
Precondiciones	↳ Que al avatar haya sido visto moviéndose o haya perdido la partida. ↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se ha cambiado el valor de la acción del avatar.

Nombre	DetectarVistoMucho ()
Responsabilidades	Comprobar si el jugador que se la liga nos está viendo movernos muchas veces y cambiar el valor del humor de acuerdo con la personalidad.
Tipo	Sistema
Referencias Cruzadas	Concepto de uso *Detectar demasiadas veces visto*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se han cambiado los valores del humor del avatar.

Nombre	DetectarDuracionPartida ()

Responsabilidades	Comprobar si la partida está siendo muy larga y cambiar el humor del avatar si es necesario.
Tipo	Sistema
Referencias Cruzadas	Concepto de uso *Detectar duración de la partida*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se han cambiado los valores del humor del avatar.

Nombre	DetectarAvatarGana ()
Responsabilidades	Detectar si el avatar ha ganado la partida.
Tipo	Sistema
Referencias Cruzadas	Concepto de uso *Detectar avatar gana el juego*.
Excepciones	
Precondiciones	
Postcondiciones	↳ Se han cambiado los valores del humor del avatar.

Nombre	DetectarAvatarPierde ()
Responsabilidades	Detectar si ha sido otro avatar el que ha ganado la partida.
Tipo	Sistema
Referencias Cruzadas	Concepto de uso *Detectar avatar pierde el juego*.
Excepciones	
Precondiciones	
Postcondiciones	↳ Se han cambiado los valores del humor del avatar.

Nombre	Contar ()
Responsabilidades	Indicar al avatar que se vuelva hacia la pared para contar.
Tipo	Sistema
Referencias Cruzadas	Concepto de uso *Volverse hacia la pared para contar*.
Excepciones	
Precondiciones	↳ Que el avatar se la ligue. ↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se ha cambiado el valor de la acción del avatar.

Nombre	DetectarContando ()
Responsabilidades	Detectar que el avatar que se la liga se ha vuelto hacia la pared para contar.
Tipo	Sistema
Referencias Cruzadas	Concepto de uso *Detectar que el que se la liga está contando*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	

Nombre	MirarJugadores ()
Responsabilidades	Darle la vuelta al avatar que está contando para que mire si los jugadores se están moviendo.
Tipo	Sistema
Referencias Cruzadas	Concepto de uso *Volverse hacia los jugadores*.
Excepciones	
Precondiciones	↳ Que el avatar se la ligue. ↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	↳ Se ha cambiado el valor de la acción del avatar.

Nombre	ElegirSpeaker ()
Responsabilidades	Decidir quién es el próximo avatar que se la liga.
Tipo	Sistema
Referencias Cruzadas	Concepto de uso *Decidir quién se la liga*.
Excepciones	
Precondiciones	↳ Que el avatar no esté realizando ninguna otra acción.
Postcondiciones	

Nombre	DetectarObstaculo ()
Responsabilidades	Detectar si hay algún obstáculo que impide el movimiento por el EV.
Tipo	Sistema
Referencias Cruzadas	Concepto de uso *Detectar obstáculo*.
Excepciones	
Precondiciones	↳ Que el avatar esté andando o corriendo.
Postcondiciones	

1.5. GLOSARIO

Al finalizar la etapa de análisis es el momento de presentar los nuevos términos que deben ser incluidos en el glosario que se viene realizando durante el desarrollo.

Término	Categoría	Descripción
Aburrirse ()	*Operación*	Aumenta el grado de aburrimiento del avatar y mostrarlo externamente.
Agredir ()	*Operación*	Hace que el avatar realice la acción de agredir al otro avatar.
Bailar ()	*Operación*	Hace que el avatar realice la acción de bailar.
Cerebro	*Concepto*	Se encarga de procesar la información que el avatar recibe a través de los sentidos y produce una reacción. Además, si es necesario, cambia el humor del avatar.
Contar ()	*Operación*	Indica al avatar que se vuelva hacia la pared para contar.
DetectarAgresion ()	*Operación*	Detecta que otro avatar me está agrediendo.
DetectarAvatarAburrido ()	*Operación*	Detecta si hay algún avatar que se esté aburriendo y obra de acuerdo con el humor y la personalidad.

DetectarAvatarConocido ()	*Operación*	Comprueba si hay conectado algún avatar al que se conozca de otras conexiones.
DetectarAvatarEnfadado ()	*Operación*	Detecta si hay algún avatar enfadado dentro del EV y obra en consecuencia, de acuerdo con el humor y la personalidad.
DetectarAvatarGana ()	*Operación*	Detecta si el avatar ha ganado la partida.
DetectarAvatarPierde ()	*Operación*	Detecta si ha sido otro avatar el que ha ganado la partida.
DetectarAvatarTriste ()	*Operación*	Detecta si hay algún avatar triste en el EV y obra en consecuencia, de acuerdo con el humor y la personalidad.
DetectarCercaPared ()	*Operación*	Detecta si el avatar está cerca de la pared y reacciona de acuerdo con la personalidad y el humor.
DetectarContando ()	*Operación*	Detecta que el avatar que se la liga se ha vuelto hacia la pared para contar.
DetectarDuracionPartida ()	*Operación*	Comprueba si la partida está siendo muy larga y cambia el humor del avatar si es necesario.
DetectarGenteAlrededor ()	*Operación*	Detecta si hay gente cerca y reacciona de acuerdo a la personalidad y al humor.
DetectarHacerReir ()	*Operación*	Detecta si otro avatar me está intentando hacer reír y me río si lo consigue.
DetectarJugadoresMoviendose()	*Operación*	Ordena a los jugadores que se estén moviendo que vuelvan a la línea de comienzo.
DetectarLejosPared ()	*Operación*	Comprueba si el avatar está lejos de la pared y reacciona de acuerdo con la personalidad y el humor.
DetectarObstaculo ()	*Operación*	Detecta si hay algún obstáculo que impide el movimiento por el EV.
DetectarOtrosCercaPared ()	*Operación*	Detecta si hay otros avatares más cerca de la pared y obra de acuerdo con los valores del humor y de la personalidad.
DetectarSaludo ()	*Operación*	Detecta si otro avatar nos está saludando y responde a su saludo si se considera necesario.
DetectarVisto ()	*Operación*	Detecta que el avatar que se la liga me ha visto moviéndome.
DetectarVistoMucho ()	*Operación*	Comprueba si el jugador que se la liga nos está viendo movernos muchas veces y cambia el valor del humor de acuerdo con la personalidad.
ElegirSpeaker ()	*Operación*	Decide quién es el próximo avatar que se la liga.
Enfadarse ()	*Operación*	Muestra el estado de enfado de un avatar.
HacerReir ()	*Operación*	Intenta provocar la risa a otro avatar.
Llorar ()	*Operación*	Hace que el avatar realice la acción de llorar.
MirarJugadores ()	*Operación*	Le da la vuelta al avatar que está contando para que mire si los jugadores se están moviendo.
Oido	*Concepto*	Refleja el sentido del oído del avatar; le capacita para reaccionar ante estímulos auditivos.
Oido.agudeza	*Atributo*	Determina el volumen del sonido que es capaz de percibir el avatar.

Patalear ()	*Operación*	Hace que el avatar realice la acción de tirarse al suelo y patalear.
PonerseTriste ()	*Operación*	Hace que el avatar muestre que está triste.
Reir ()	*Operación*	Hace que mi avatar se ría.
SaltarAlegria ()	*Operación*	Hace que el avatar realice el gesto de saltar de alegría.
Saludar ()	*Operación*	Saluda a los avatares que se encuentren conectados y aparezcan dentro de la lista de amigos.
Sentidos	*Concepto*	Son el medio a través del cual el avatar percibe lo que sucede en el EV.
Vista	*Concepto*	Refleja el sentido de la vista del avatar en el EV. Le da la capacidad de reaccionar ante estímulos visuales.
Vista.angulo_vision	*Atributo*	Ángulo de visión del avatar.
Vista.radio_vision	*Atributo*	Distancia hasta la cual un avatar es capaz de distinguir algo con la vista.
VolverLineaPartida ()	*Operación*	Hace que el avatar vuelva a la línea de comienzo.

2. DISEÑO

Para realizar el diseño de este segundo ciclo, se van a seguir también las recomendaciones definidas por Larman en su método [Larman99], pero además, siguiendo con lo que ya se ha hecho en la etapa de análisis, se va a completar con la introducción de un nuevo apartado en el que se desarrollarán una serie de diagramas de estados a partir del que se presentó en la fase de análisis. El motivo de ello es que, a pesar de que Larman desaconseja su utilización, aunque la permite, creemos que en este caso en concreto es la mejor manera para modelar el comportamiento de los avatares.

2.1. DEFINICIÓN DE LA INTERFAZ DE USUARIO

Según lo que se ha estado definiendo hasta el momento, cabría pensar que en este punto es necesario introducir algún mecanismo para que el usuario pueda delegar ciertas funciones en su avatar. Sin embargo, debido a la complejidad del sistema a construir, lo que se va a hacer en su lugar es que las acciones que puede hacer un usuario con su avatar van a venir determinadas por el avatar que escoja, de manera que cada un de ellos tendrá un grado de control distinto sobre las acciones que pueden realizar.

De esta manera, no es necesario cambiar o ampliar el interfaz que se ha definido en el primer ciclo de desarrollo, pues nos permitirá seguir manejando todas las funcionalidades del sistema.

2.2. DIAGRAMAS DE ESTADO

Como se comentaba en la introducción a la etapa de diseño, los diagramas de estado son un instrumento que, a pesar de estar definido en la especificación de UML, no es algo con lo que Larman aconseje trabajar, ya que considera que se puede hacer uso de otras herramientas, como los diagramas de colaboración, que reúnen toda la información que se puede encontrar en un diagrama de estados.

Si bien esto es cierto, en un sistema en el que es importante el estado de un elemento, los diagramas de estados presentan dos grandes ventajas. Por un lado, dan una visión más clara de cómo se producen los cambios de estado, y a todo lo que pueda ser útil para aclarar el funcionamiento de una aplicación debe sacársele todo el partido posible. Por otro lado, aunque en los diagramas de colaboración pueda encontrarse toda

la información necesaria, lo más normal es que esa información se encuentre dispersa entre varios diagramas, y además será algo accesorio al proceso que se describe dentro del diagrama de colaboración; en contrapartida, en un diagrama de estados se encontrará toda la información que deseemos conocer respecto a un aspecto concreto de los diferentes estados del elemento representado en el diagrama, lo cual evitará que se omitan cosas que de otra manera se podrían olvidar o podrían pasar desapercibidas.

Es sobre todo por estas dos razones por lo que se ha decidido introducir los diagramas de estados en esta fase de diseño.

2.2.1. Ganar

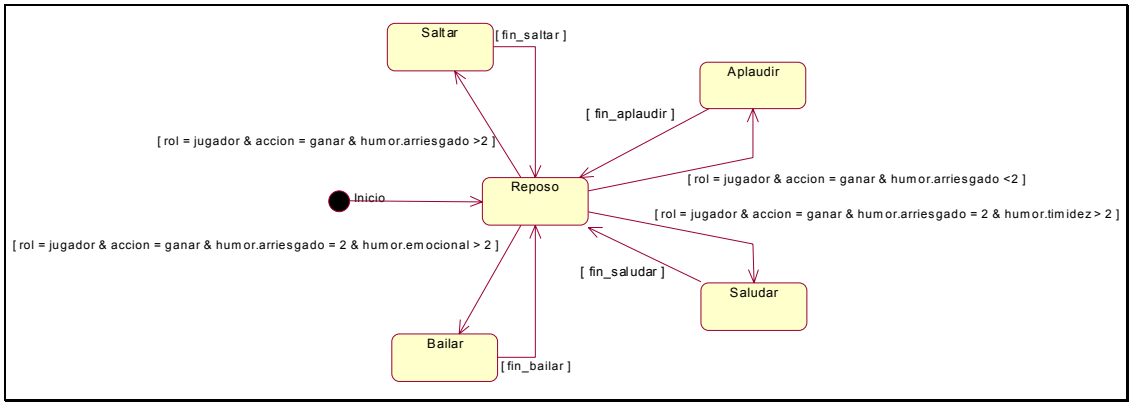

Figura 111: Ganar

2.2.2. Perder

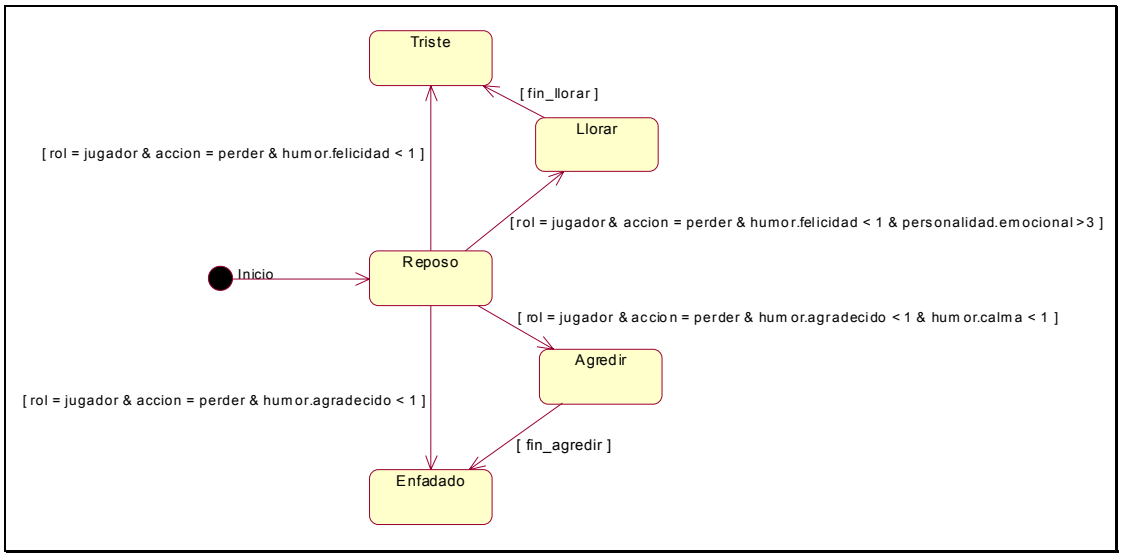

Figura 112: Perder

2.2.3. Visto

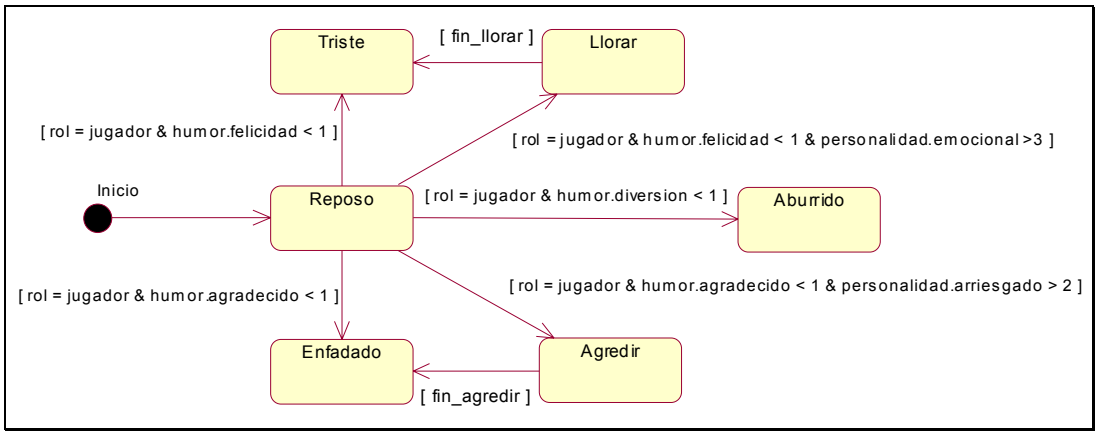

Figura 113: Visto

2.2.4. Reir

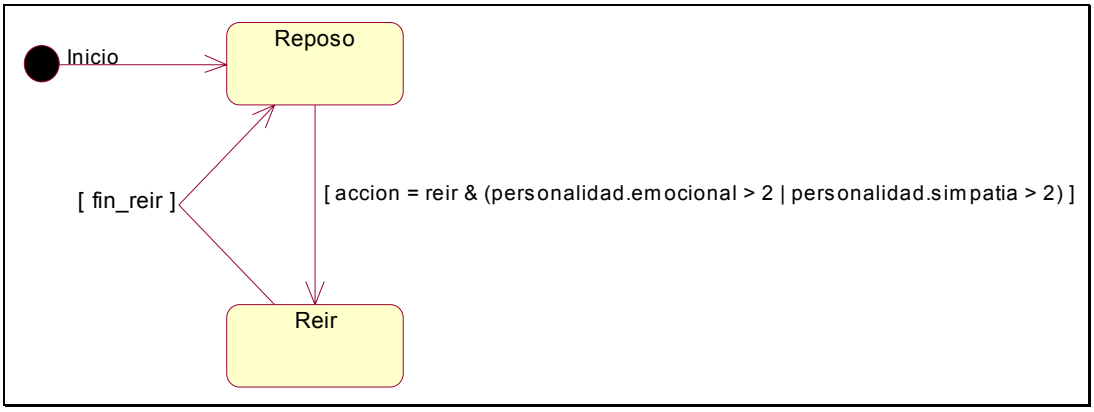

Figura 114: Reir

2.2.5. Moverse

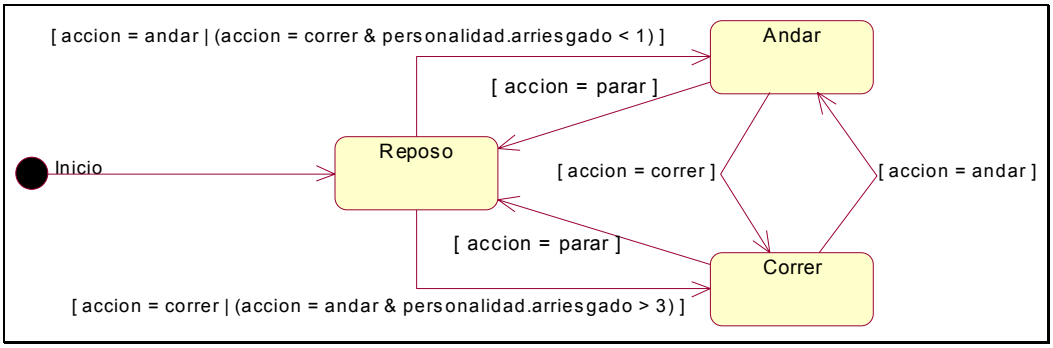

Figura 115: Moverse

2.3. DIAGRAMAS DE INTERACCIÓN

Para la realización de los diagramas de interacción en este segundo ciclo de desarrollo, se ha tomado la decisión de utilizar la representación en forma de diagramas de secuencia. A pesar de la preferencia expresada por Larman sobre la utilización de diagramas de colaboración [Larman99], los motivos para la utilización de los diagramas de secuencia son que, por un lado, nos van a dar, en este caso, más expresividad que los de colaboración, y por otro, que va a interesar más la visión que aportan los diagramas de secuencia que la de los diagramas de colaboración.

2.3.1. DetectarAvatarConocido ()

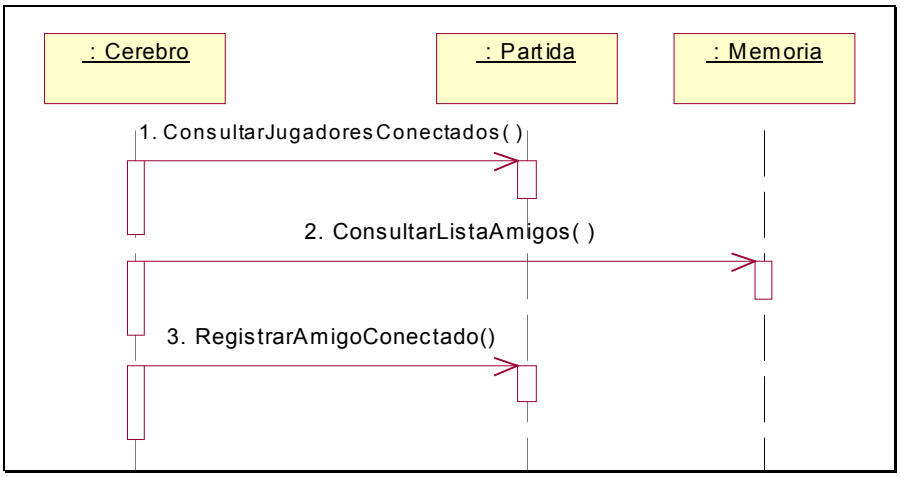

Figura 116: DetectarAvatarConocido ()

2.3.2. Saludar ()

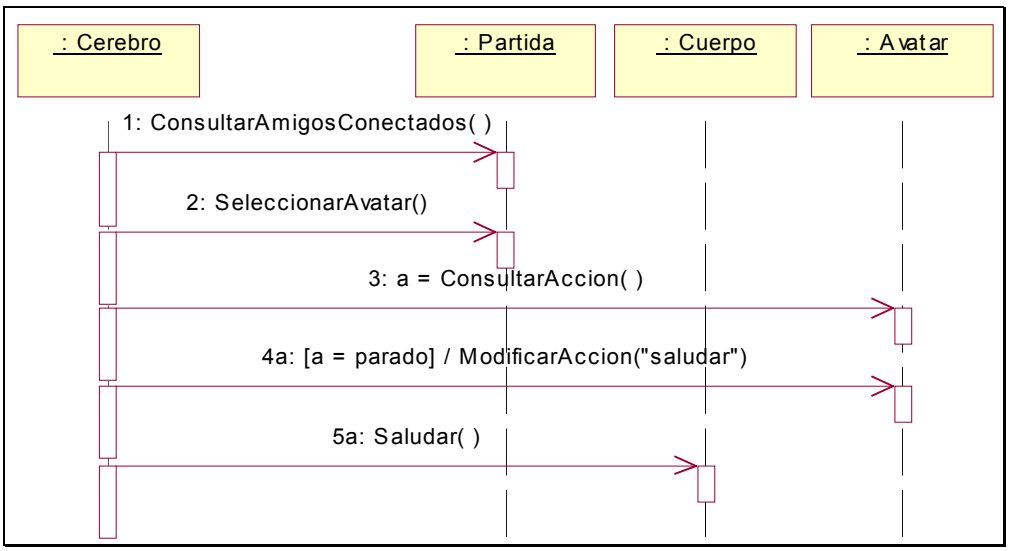

Figura 117: Saludar ()

2.3.3. DetectarSaludo ()

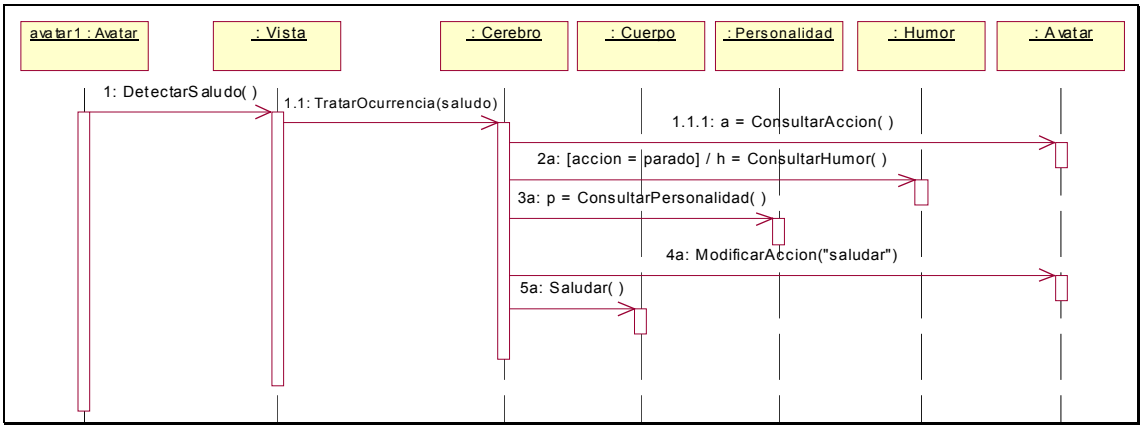

Figura 118: DetectarSaludo ()

2.3.4. HacerReir ()

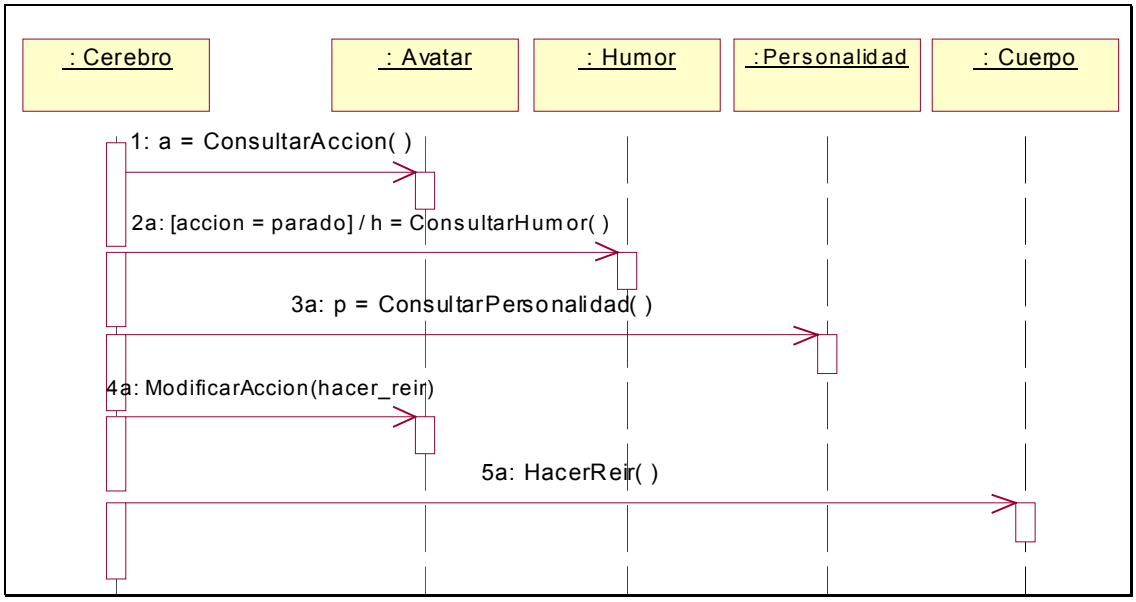

Figura 119: HacerReir ()

2.3.5. DetectarHacerReir ()

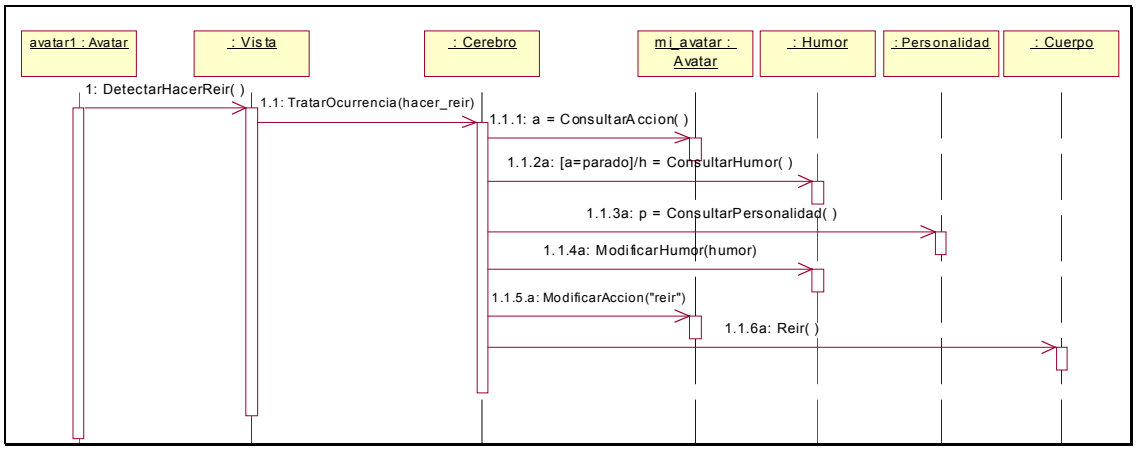

Figura 120: DetectarHacerReir ()

2.3.6. Reir ()

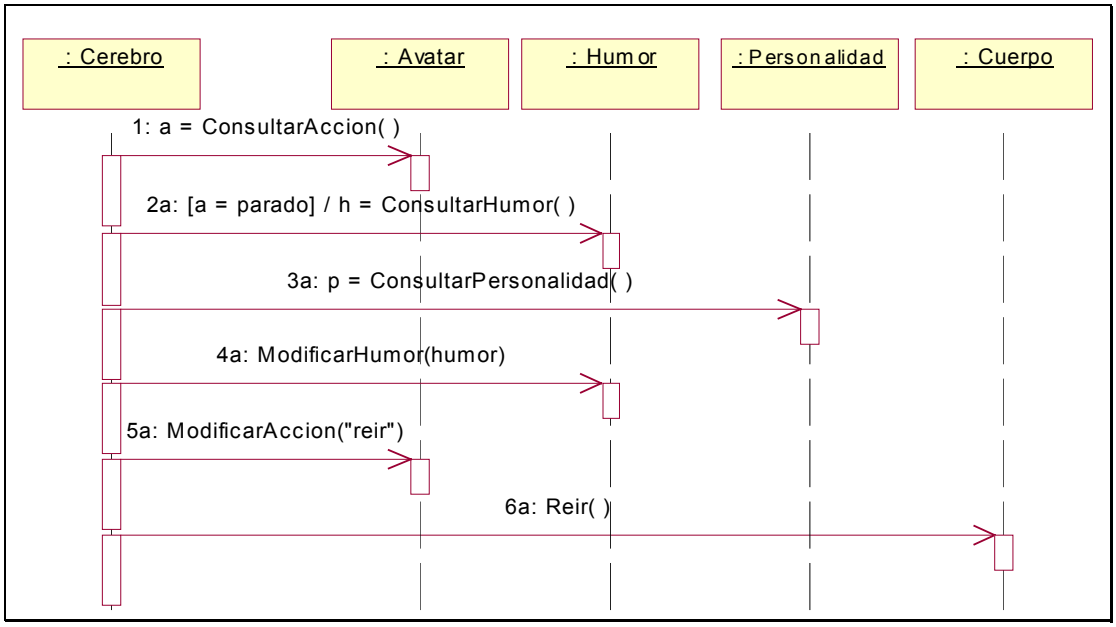

Figura 121: Reir ()

2.3.7. SaltarAlegria ()

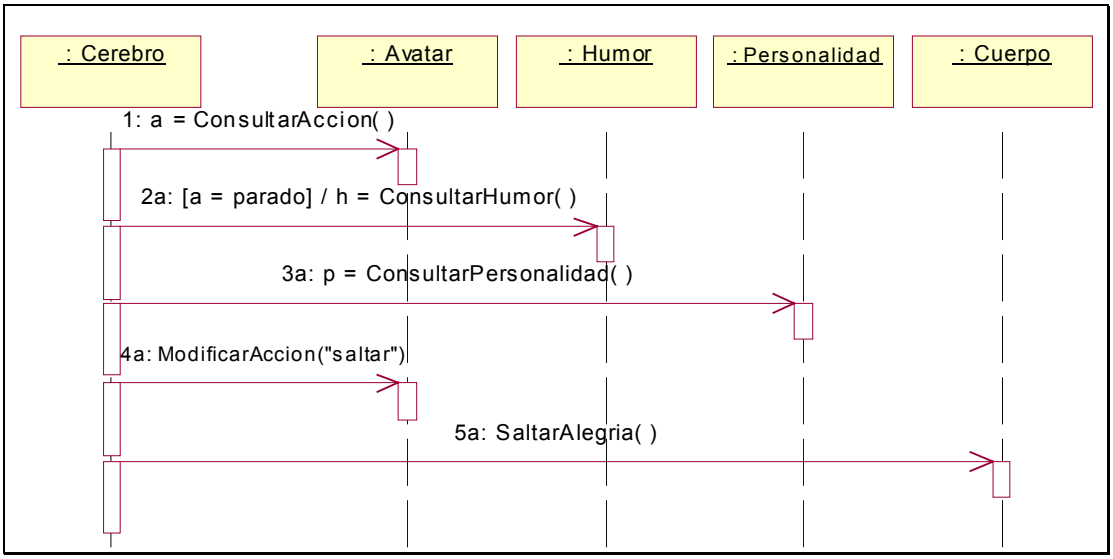

Figura 122: SaltarAlegria ()

2.3.8. Bailar ()

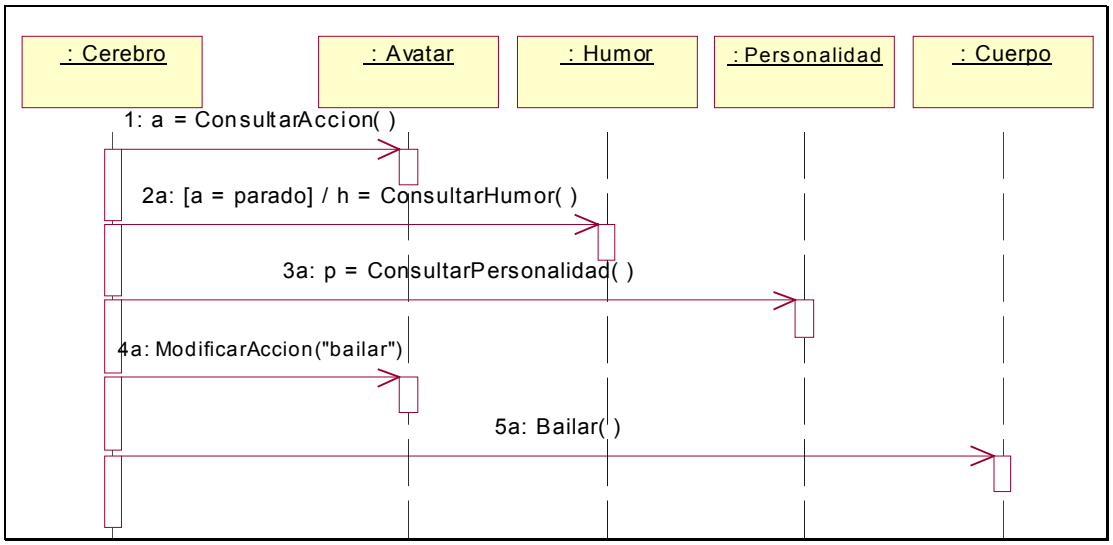

Figura 123: Bailar ()

2.3.9. PonerseTriste ()

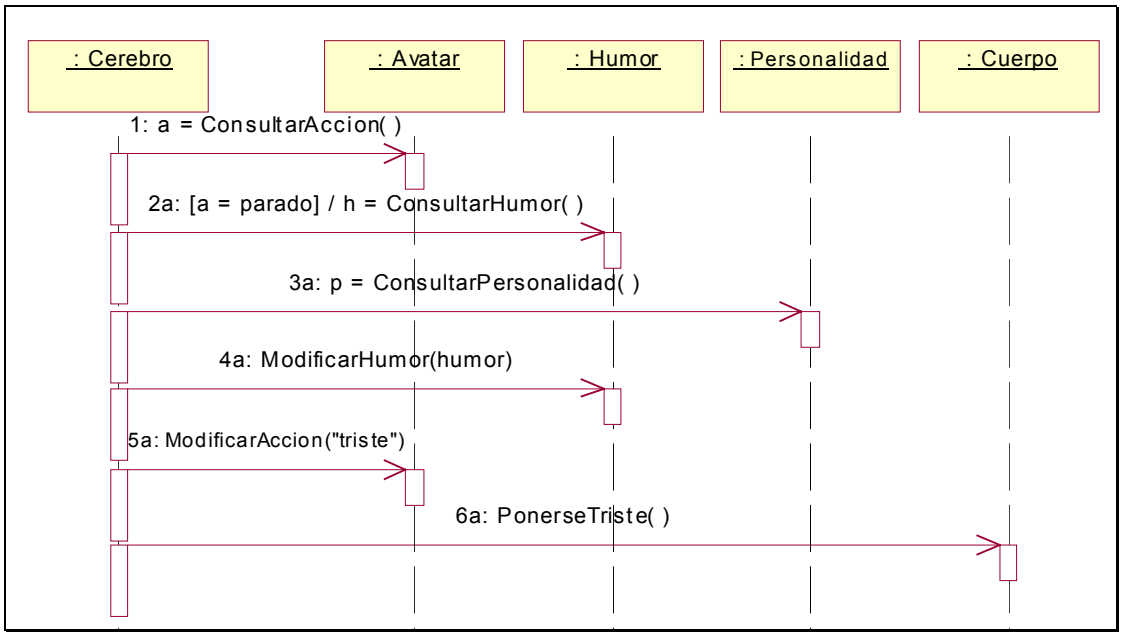

Figura 124: PonerseTriste ()

2.3.10. DetectarAvatarTriste ()

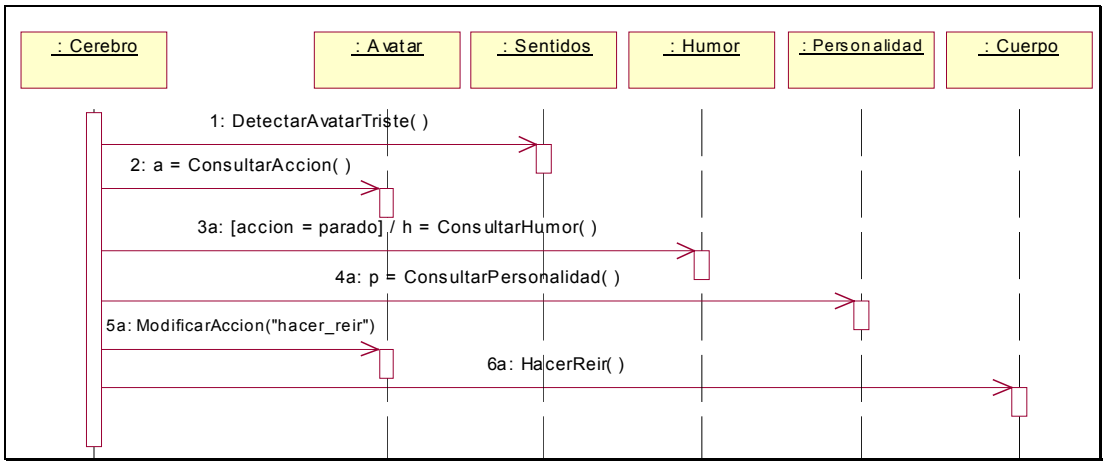

Figura 125: DetectarAvatarTriste ()

2.3.11. Llorar ()

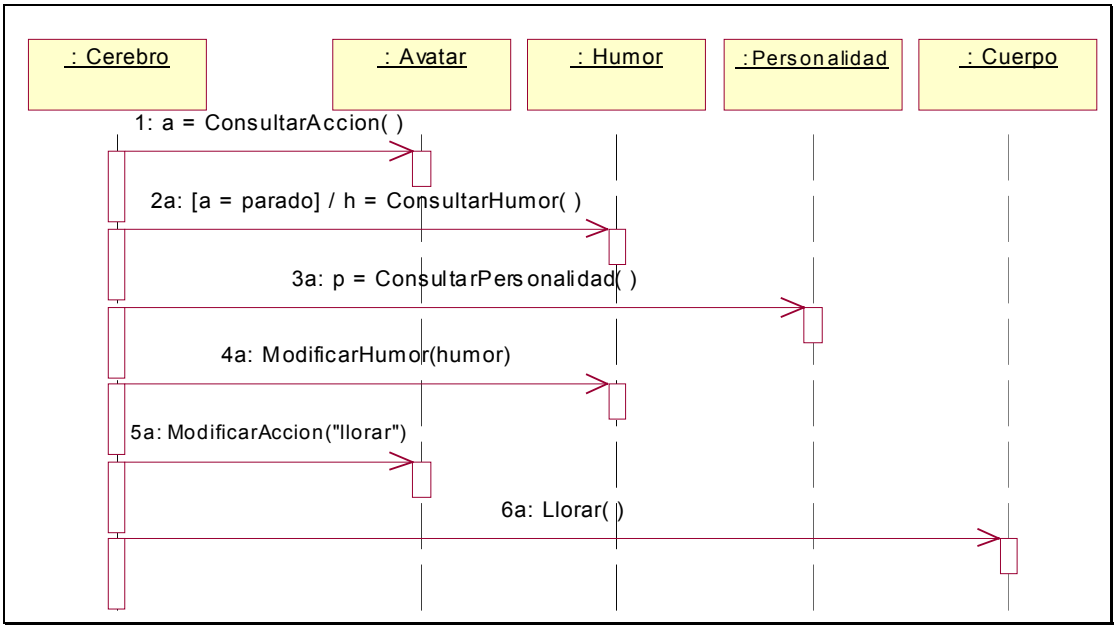

Figura 126: Llorar ()

2.3.12. Enfadarse ()

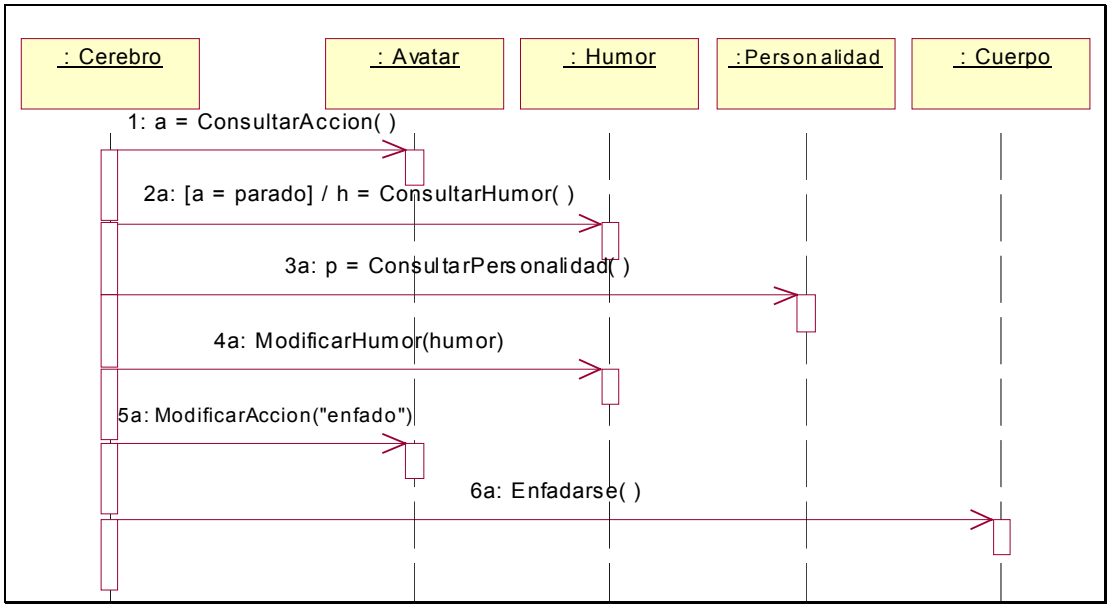

Figura 127: Enfadarse ()

2.3.13. DetectarAvatarEnfadado ()

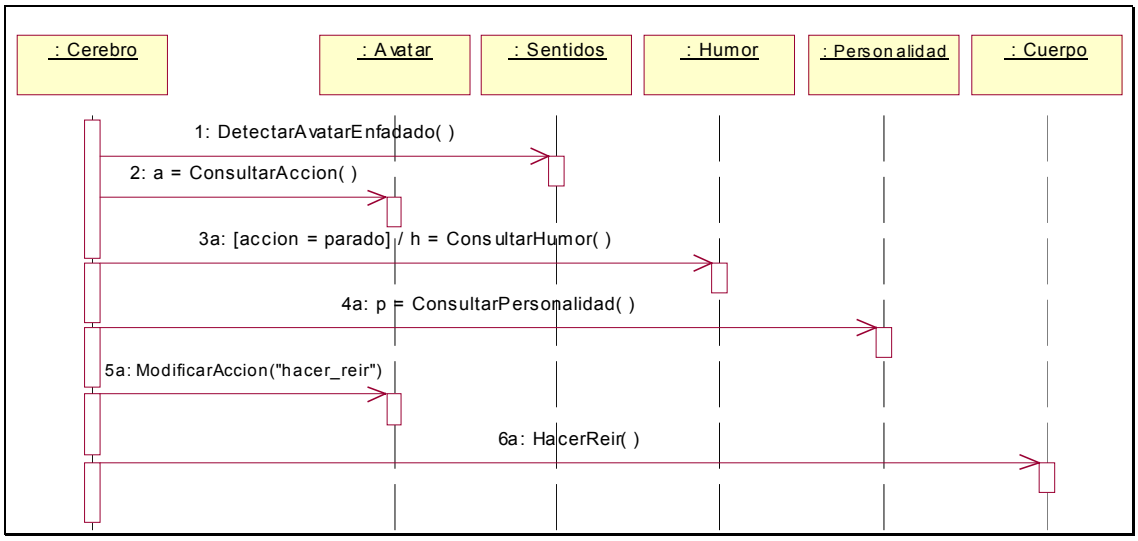

Figura 128: DetectarAvatarEnfadado ()

2.3.14. Patalear ()

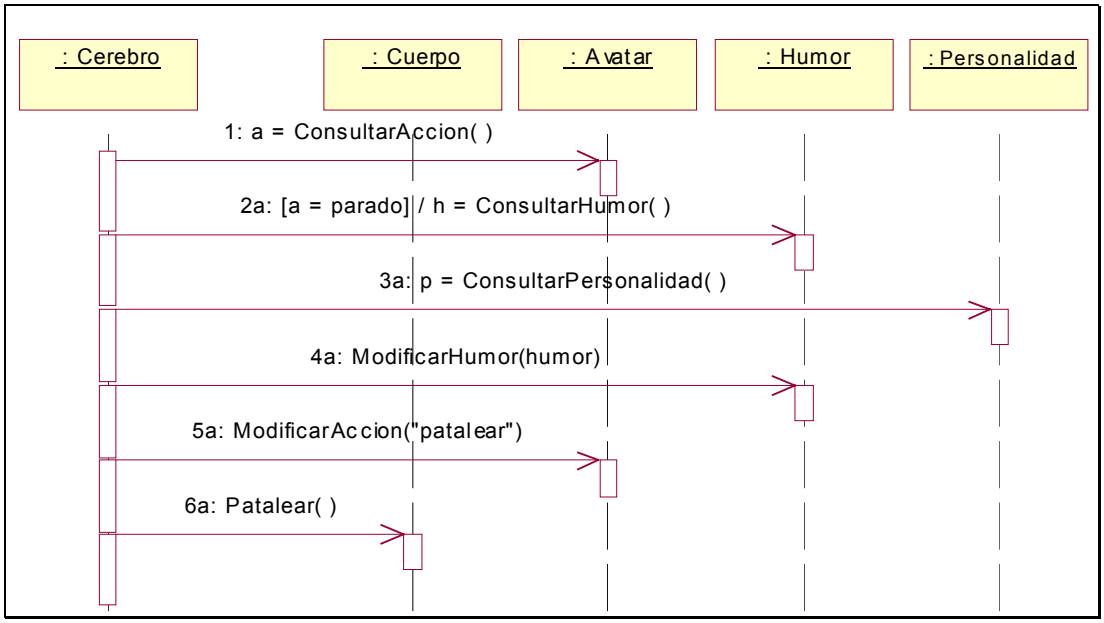

Figura 129: Patalear ()

2.3.15. Agredir ()

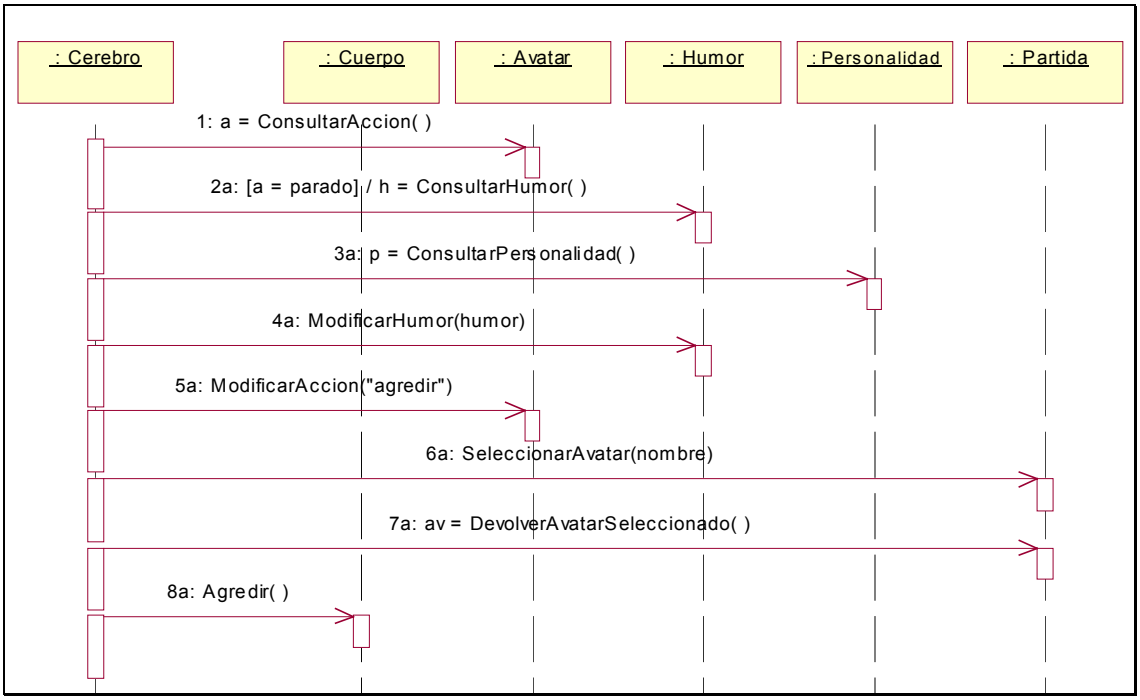

Figura 130: Agredir ()

2.3.16. DetectarAgresion ()

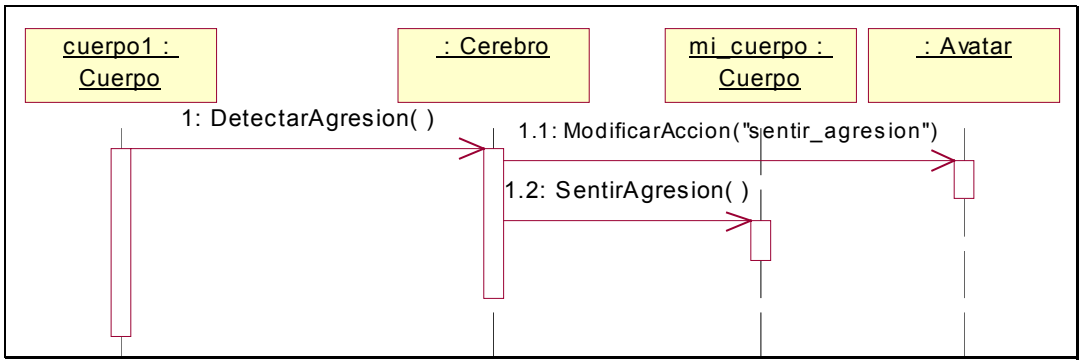

Figura 131: DetectarAgresión ()

2.3.17. Aburrirse ()

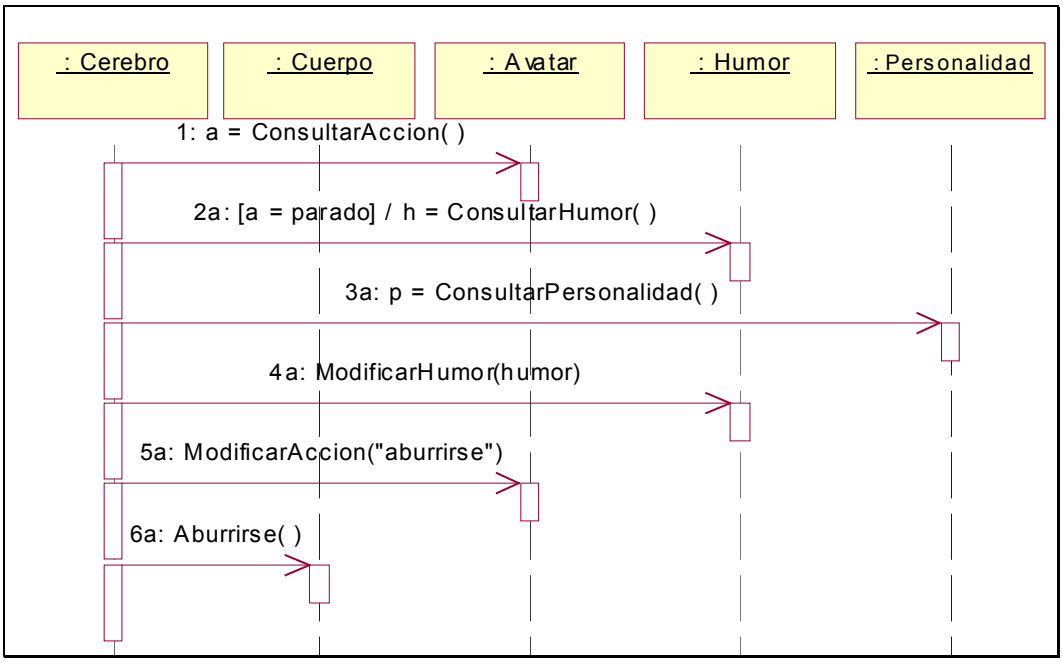

Figura 132: Aburrirse ()

2.3.18. DetectarAvatarAburrido ()

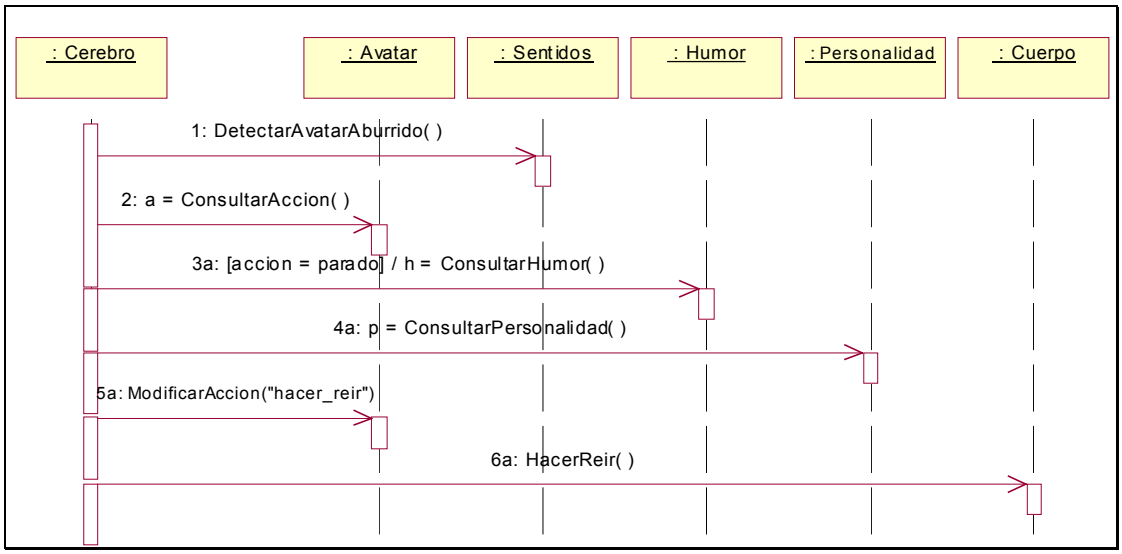

Figura 133: DetectarAvatarAburrido ()

2.3.19. DetectarGenteAlrededor ()

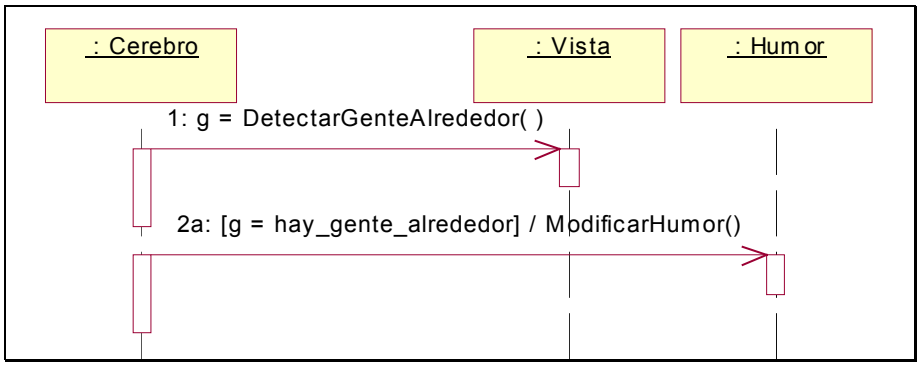

Figura 134: DetectarGenteAlrededor ()

2.3.20. DetectarLejosPared ()

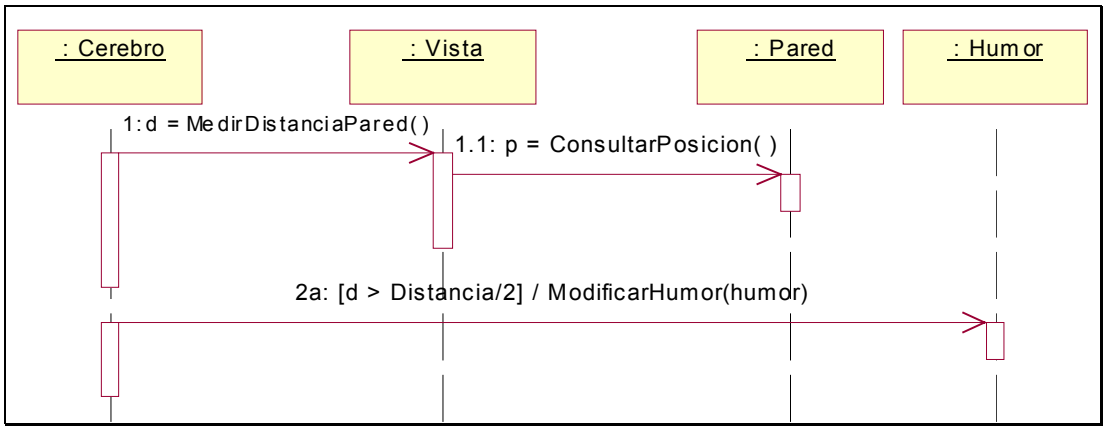

Figura 135: DetectarLejosPared ()

2.3.21. DetectarCercaPared ()

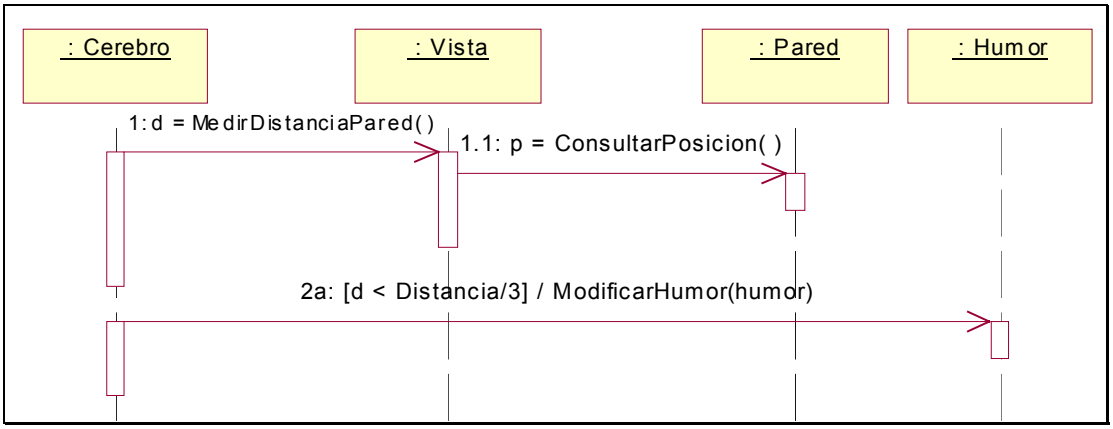

Figura 136: DetectarCercaPared ()

2.3.22. DetectarOtrosCercaPared ()

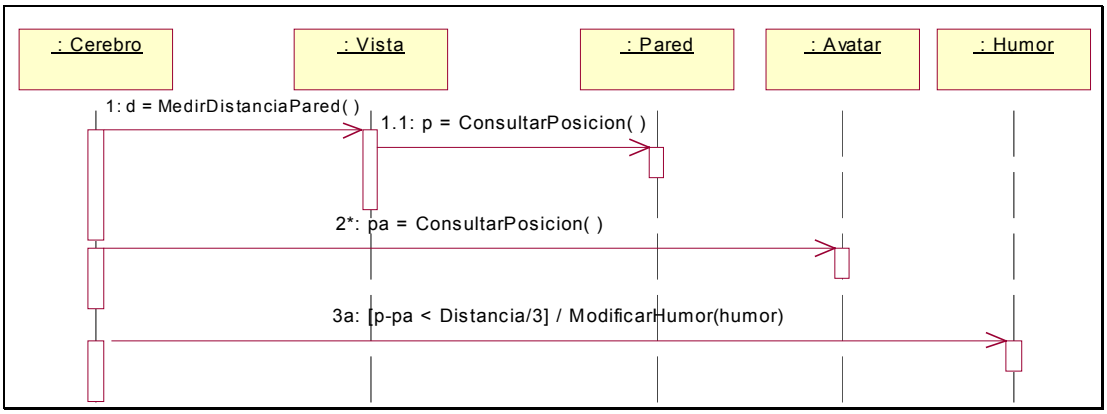

Figura 137: DetectarOtrosCercaPared ()

2.3.23. DetectarJugadoresMoviendose ()

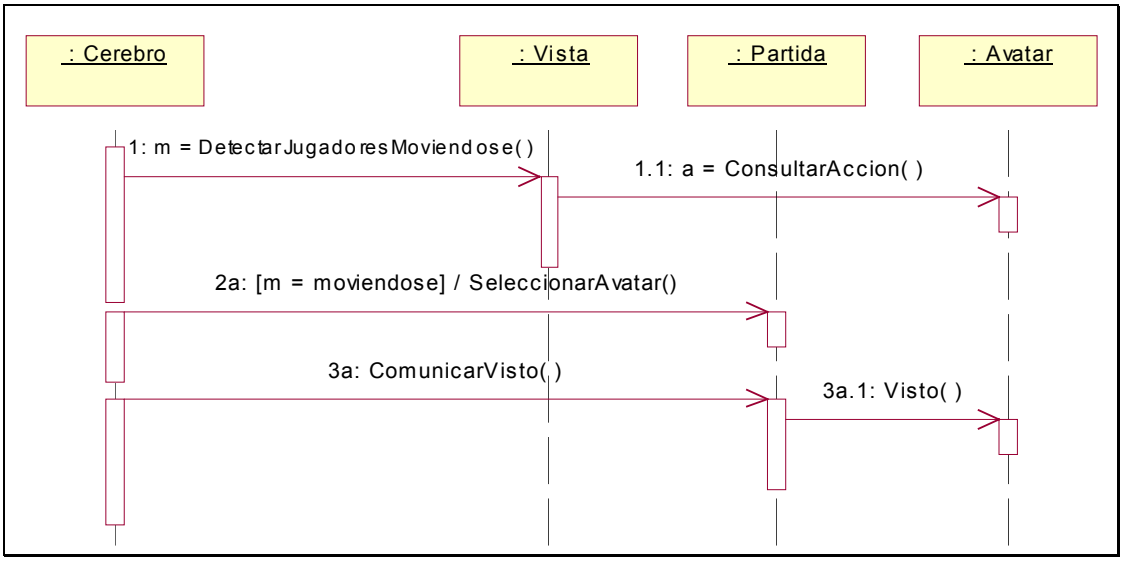

Figura 138: DetectarJugadoresMoviendose ()

2.3.24. DetectarVisto ()

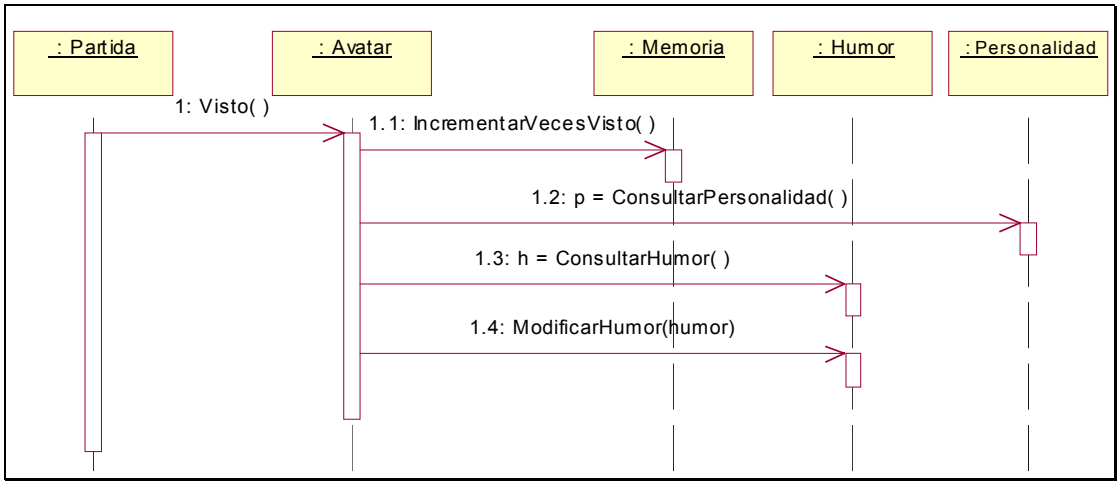

Figura 139: DetectarVisto ()

2.3.25. VolverLineaPartida ()

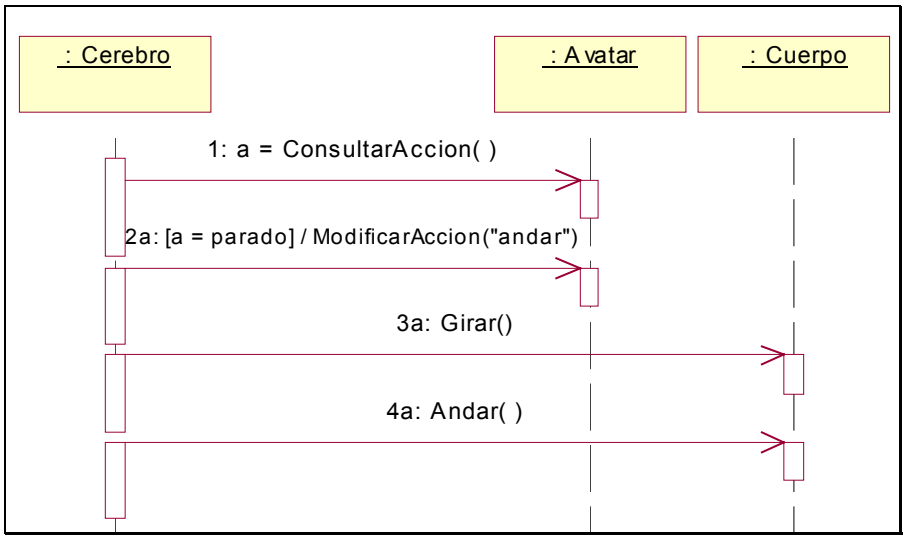

Figura 140: VolverLineaPartida ()

2.3.26. DetectarVistoMucho ()

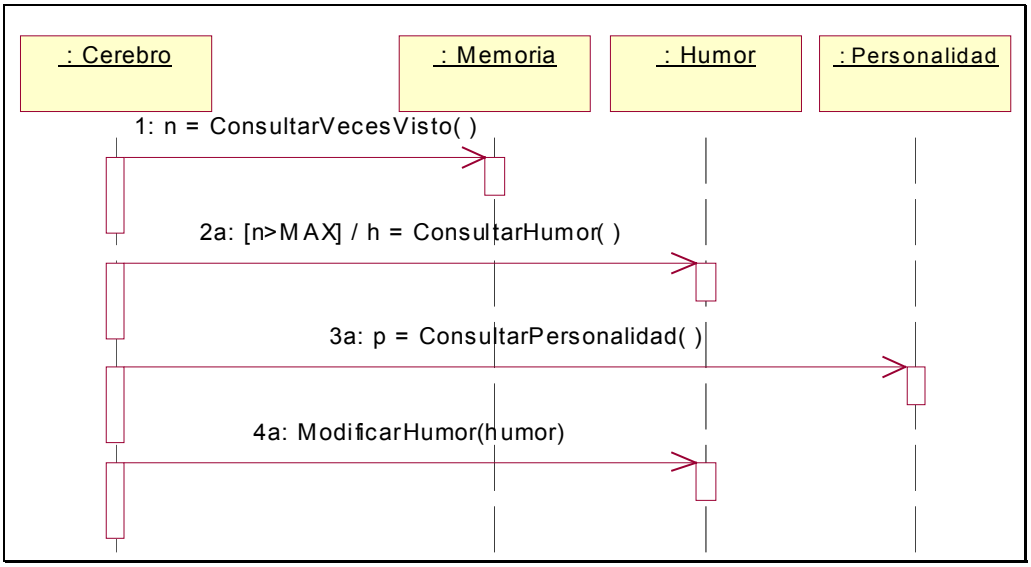

Figura 141: DetectarVistoMucho ()

2.3.27. DetectarDuracionPartida ()

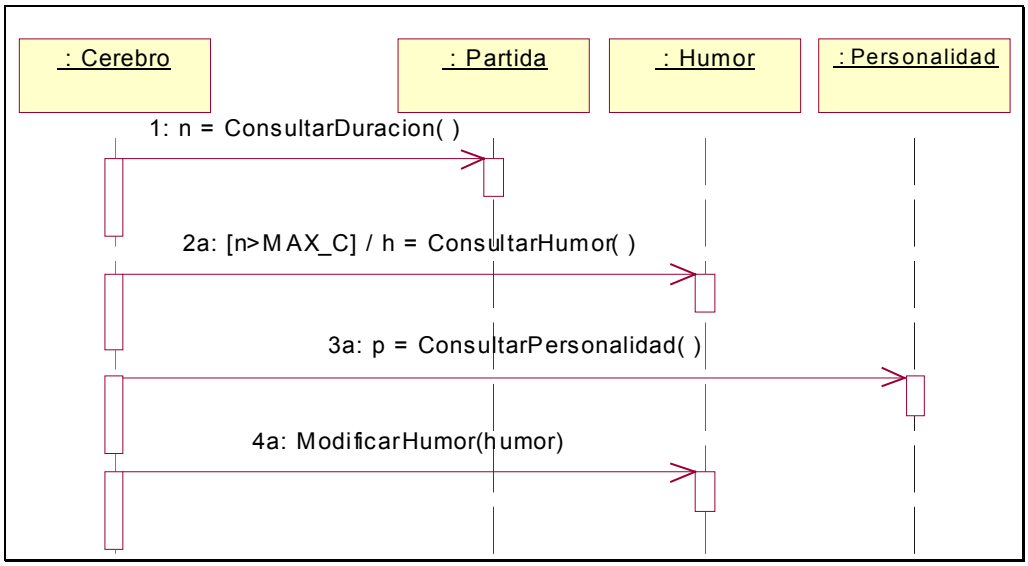

Figura 142: DetectarDuracionPartida ()

2.3.28. DetectarAvatarGana ()

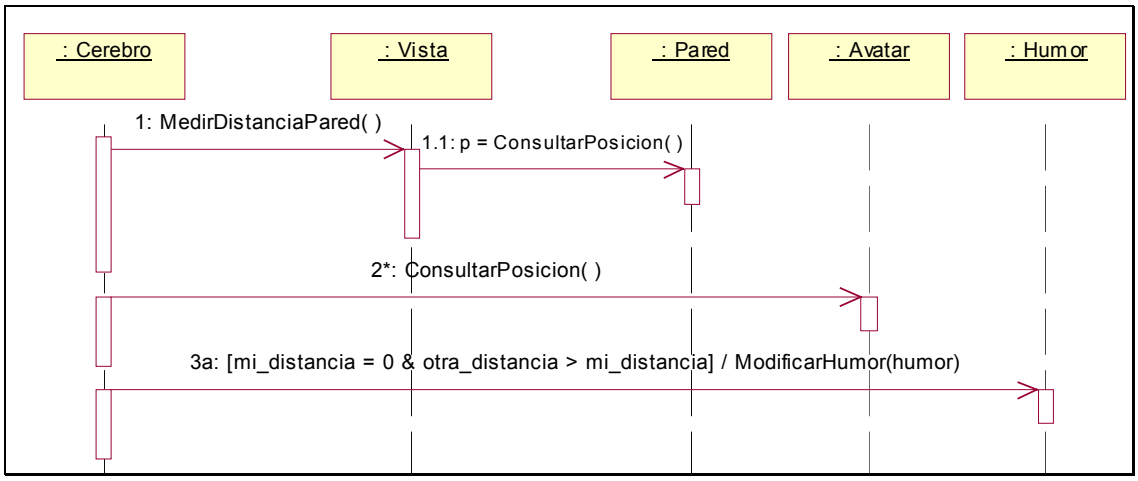

Figura 143: DetectarAvatarGana ()

2.3.29. DetectarAvatarPierde ()

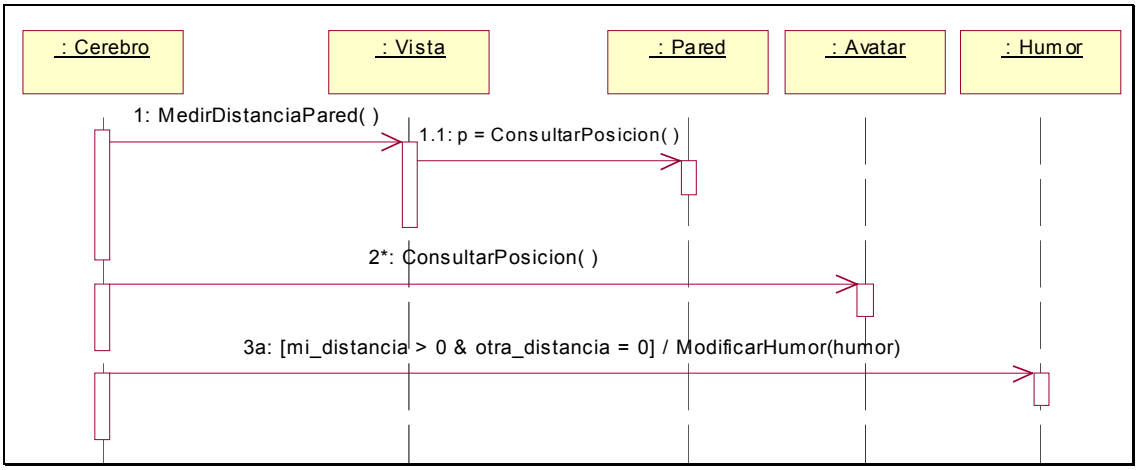

Figura 144: DetectarAvatarPierde ()

2.3.30. Contar ()

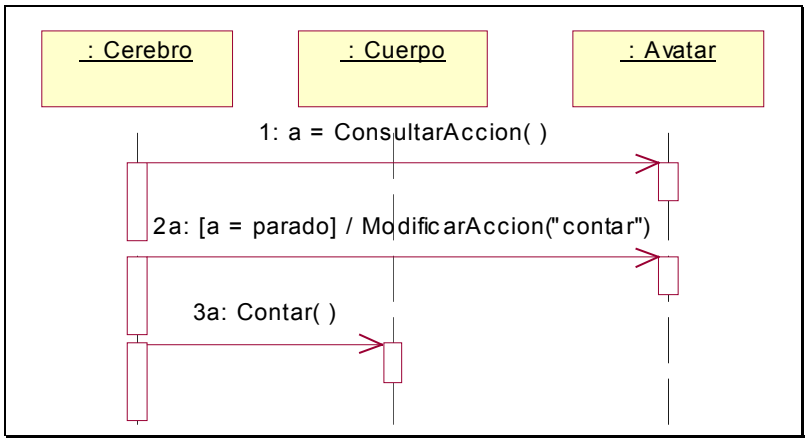

Figura 145: Contar ()

2.3.31. DetectarContando ()

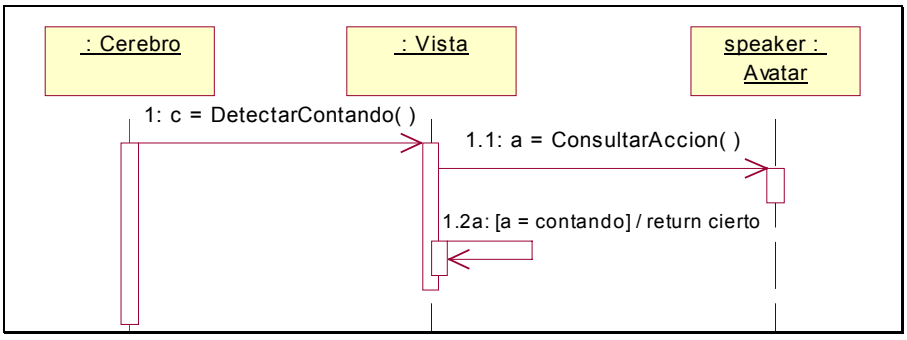

Figura 146: DetectarContando ()

2.3.32. MirarJugadores ()

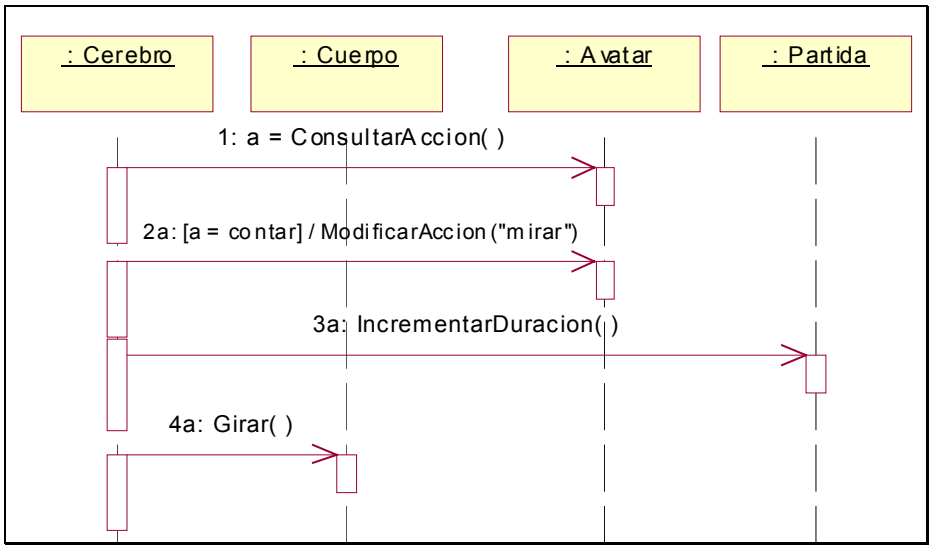

Figura 147: MirarJugadores

2.3.33. ElegirSpeaker ()

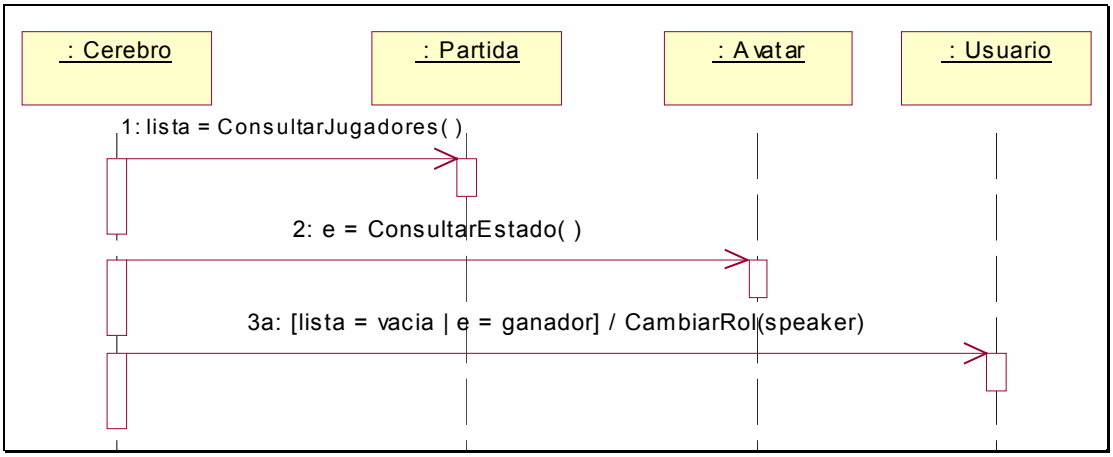

Figura 148: ElegirSpeaker ()

2.3.34. DetectarObstaculo ()

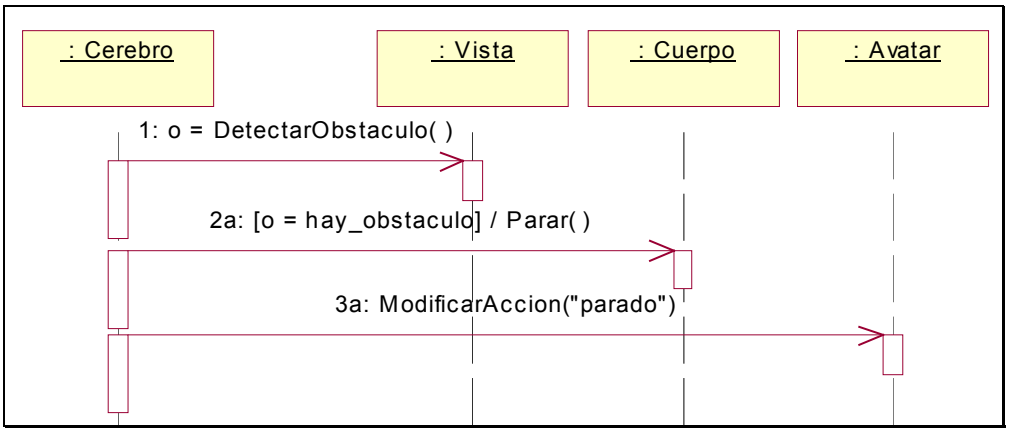

Figura 149: DetectarObstaculo ()

2.4. DIAGRAMA DE CLASES DE DISEÑO

Se amplía el diagrama del ciclo anterior con la información obtenida de los diagramas de estados y de interacción. Los cambios más significativos son la introducción de tres nuevas clases (Sentidos, Vista y Oido) y la ampliación de las operaciones existentes para la clase Partida.

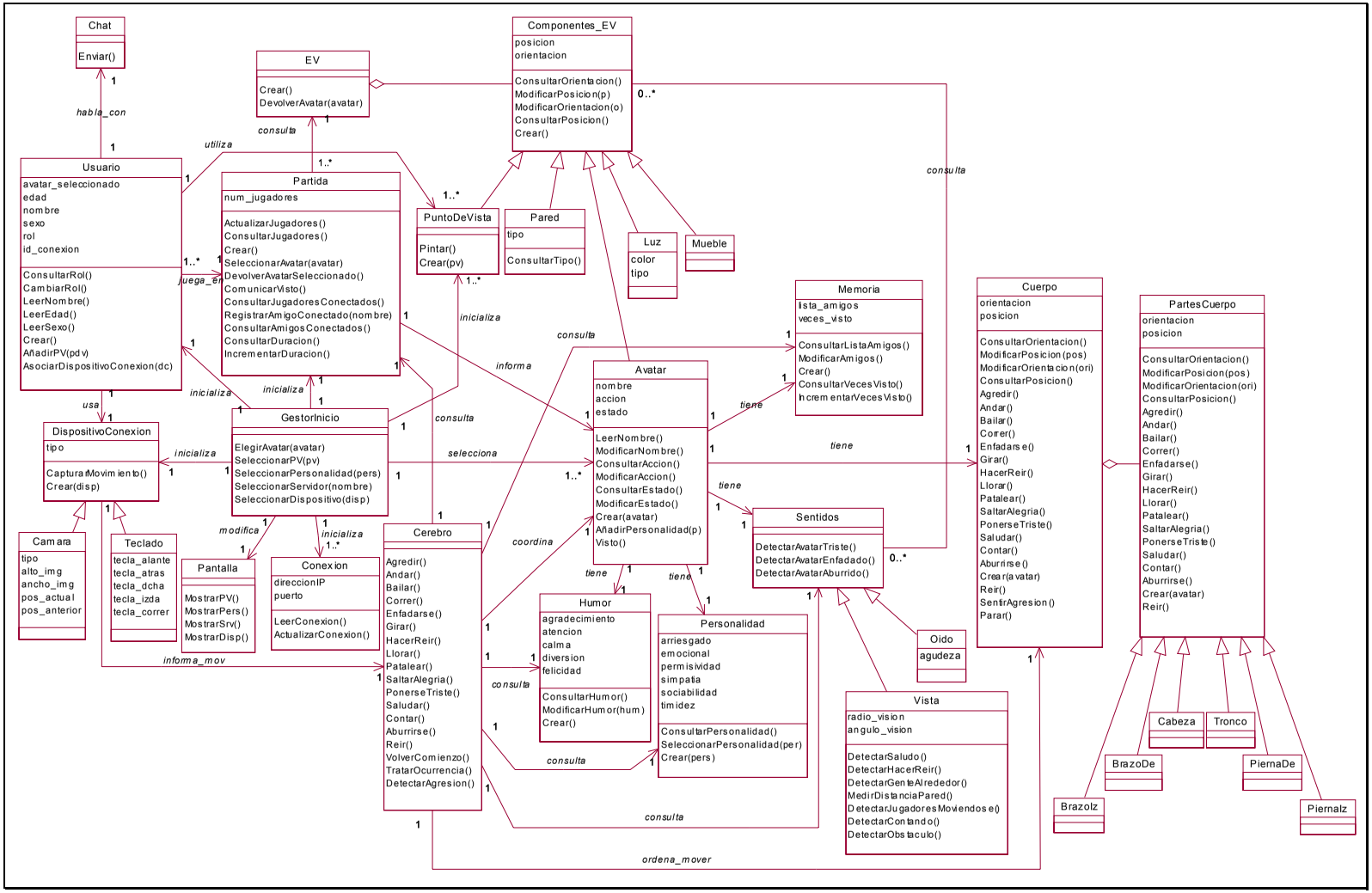

Figura 150: Ampliación del diagrama de clases de diseño

2.5. MODELO DE DATOS

En lo referente al modelos de datos, no se ha encontrado nada que haga pensar que se desea guardar información adicional, razón por la cual no se realizará ningún cambio al modelo planteado en el anterior ciclo de desarrollo.

2.6. GLOSARIO

Ampliamos el glosario con los nuevos elementos aparecidos en el diseño.

Término	Categoría	Descripción
Oido.Agudeza	*Atributo*	Representa el umbral mínimo a partir del cual el oído es capaz de percibir un sonido.
Vista.Radio_vision	*Atributo*	Distancia máxima a la que se puede ver algo.
Vista.angulo_vision	*Atributo*	Ángulo de visión.
Oido	*Clase*	Representa el sentido del oído, con el cual se pueden percibir ciertos sucesos del EV, como que un avatar esté llorando.
Sentidos	*Clase*	Representa los sentidos del avatar para percibir lo que sucede en el EV.
Vista	*Clase*	Representa el sentido de la vista del avatar para ver lo que sucede en el EV.
Sentidos.DetectarAvatarAburrido ()	*Método*	Explora el EV en busca de avatares aburridos.
Sentidos.DetectarAvatarEnfadado ()	*Método*	Explora el EV en busca de avatares enfadados.
Sentidos.DetectarAvatarTriste ()	*Método*	Explora el EV en busca de avatares tristes.
Vista.DetectarContando ()	*Método*	Comprueba si el jugador que se la liga está contando.
Vista.DetectarGenteAlrededor ()	*Método*	Explora el EV para ver si tenemos avatares cerca.
Vista.DetectarHacerReir ()	*Método*	Con esta operación, la vista es capaz de detectar que otro avatar nos está intentando hacer reír.
Vista.DetectarJugadoresMoviendose()	*Método*	La utiliza el jugador que se la liga para ver si algún jugador se está moviendo.
Vista.DetectarObstaculo ()	*Método*	Comprueba si hay obstáculos que nos impidan desplazarnos.
Vista.DetectarSaludo ()	*Método*	Con esta operación, la vista es capaz de detectar que otro avatar nos está saludando.
Vista.MedirDistanciaPared ()	*Método*	Comprueba la distancia que nos separa de la pared del fondo.

Cerebro.TratarOcurrencia ()	*Método*	Cuando los sentidos detectan algo en el EV, se lo comunican al cerebro con esta operación para que decida qué hacer.
Cerebro.DetectarAgresion ()	*Método*	Detecta si algún avatar nos agrede.
Cuerpo.Parar ()	*Método*	Detiene el movimiento del avatar.
Cuerpo.SentirAgresion ()	*Método*	Cuando algún avatar nos agrede, muestra el resultado de la agresión.
Partida.ConsultarJugadoresConectados ()	*Método*	Devuelve una lista de los jugadores que hay conectados al EV.
Partida.ConsultarAmigosConectados ()	*Método*	Devuelve la lista de los amigos que se hayan conectados al EV.
Partida.RegistrarAmigoConectado(nombre)	*Método*	Apunta en una lista los amigos del jugador local.
Partida.ConsultarDuracion ()	*Método*	Devuelve el número de veces que el jugador que se la liga ha contado en la partida actual.
Partida.IncrementarDuracion ()	*Método*	Incrementa el número de veces que el jugador que se la liga ha contado durante la partida.
tiene	*Asociación*	Un Avatar tiene Sentidos para percibir lo que sucede en el EV.

3. IMPLEMENTACIÓN

La implementación de este segundo ciclo de desarrollo ha sido bastante más rápida que la del primero, ya que gran parte de los elementos que se necesitaban se encontraban implementados en este último, por lo que ha sido posible su reutilización.

Sin embargo, sí se ha debido dedicar más tiempo a la realización de pruebas para comprobar el correcto funcionamiento de las acciones que los avatares realizaban de manera autónoma, aunque, como se ha dicho, el hecho de que hubiese que implementar pocas cosas nuevas ha facilitado la práctica ausencia de errores de implementación durante esta fase.

Donde ha habido que realizar mayor número de cambios ha sido en los avatares, en los cuales se ha hecho necesario ampliar los *scripts* existentes y añadir otros nuevos, como se puede ver en la Figura 151.

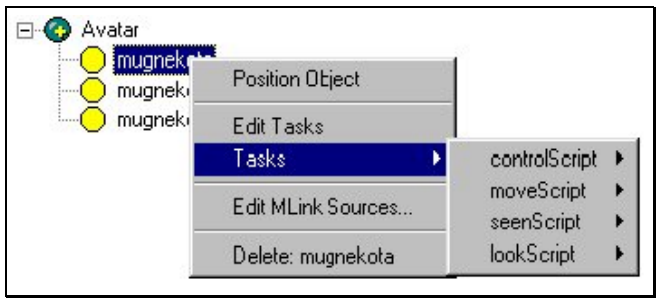

Figura 151: *Scripts* de los avatares

Los *scripts* nuevos que se han introducido han sido:

- seenScript: sólo está activo si el avatar no se la liga. Se encarga de controlar si el avatar ha sido visto moviéndose.
- lookScript: sólo está activo cuando el jugador se la liga. Se encarga de comprobar si hay algún jugador moviéndose.

Es importante señalar que no ha sido necesario elaborar ningún mecanismo especial para implementar las tareas que se han de realizar de manera cíclica y que se encuentran descritas en los conceptos de uso, ya que WorldUp, cuando se ejecuta la aplicación, va recorriendo los objetos visibles del EV para ver si puede ejecutar alguna acción codificada en los scripts que llevan asociados. Ha bastado, por tanto, la inclusión de las tareas de detección en el *script* controlScript para que éstas se realicen de manera cíclica, sin tener que preocuparse más por ello.

CONCLUSIONES Y TRABAJO FUTURO

1. CONCLUSIONES

Después de haber implementado un sistema como el que se ha visto a lo largo del presente desarrollo, y siendo como es un tipo de aplicación en el que todavía no existe demasiada experiencia, se hace necesaria una pequeña retrospectiva para analizar todas las dificultades encontradas a lo largo del desarrollo, así como todas las herramientas que han servido de ayuda para facilitar el desarrollo.

Haciendo un recorrido por todas las fases del método de Larman, las conclusiones a las que se puede llegar son las siguientes:

- Planificación y especificación de requisitos: en esta fase, donde se describen por primera vez los casos de uso, se echa de menos alguna referencia a la manera en que se debería construir el documento de especificación de requisitos para poder sacarle partido a la hora de describir los casos de uso. Por otro lado, también sería necesario especificar algún método para describir de manera más concreta las acciones que no realice directamente el usuario, ya que, si no se hace de manera correcta y lo más exhaustiva posible, ocasiona defectos que se van arrastrando a lo largo de todo el desarrollo.

- Análisis: de igual manera que en la etapa de planificación, hace falta alguna herramienta que permita desarrollar las funcionalidades que no corresponden directamente al usuario. De igual forma en que existe un camino a seguir desde los casos de uso en formato de alto nivel hasta los contratos de operaciones, debería existir un método para llegar desde la descripción de tareas realizadas automáticamente por el sistema hasta los mismos contratos de operaciones o alguna otra cosa similar a ellos.

- Diseño: probablemente sea la etapa en la que se han encontrado más carencias a la hora de realizar el desarrollo. En primer lugar, ha habido que introducir una fase completamente nueva para poder contemplar el diseño 3D que iba a constituir el EV. Siendo como es una de las partes constituyentes del sistema más importantes, es un hecho bastante desafortunado el no poder encontrar una fase en el método de desarrollo que al menos dé la oportunidad de incluir en él este proceso. Por otro lado, a pesar de permitir su utilización, el método de Larman deja prácticamente de lado la estructura física del sistema, y eso es

algo que, en un sistema de este tipo, puede dar lugar a que se dejen cabos sueltos, ya que es necesario, en una aplicación que depende tanto de dispositivos físicos y componentes software, especificar cómo se relacionan entre ellos. También hay que destacar que, aunque probablemente se les ha sacado menos partido del que hubiese sido posible, habría que darle mayor importancia a la utilización de los diagramas de transición de estados, pues parece a todas luces la herramienta más potente para reflejar el complejo comportamiento que tendrán elementos del sistema como los avatares.

Como se puede ver, el método de Larman puede funcionar bien en la construcción de aplicaciones de corte más o menos tradicional, pero presenta grandes lagunas con sistemas que se salen de lo habitual. Aún así, no es posible negar que para las partes del sistema que se asimilan a un sistema clásico, el método de Larman permite ir haciendo un desarrollo minucioso en el que se contempla toda la información necesaria para que el sistema haga todo lo que tiene que hacer, y el hecho de que se trate de un método orientado a objetos posibilita que se adapte muy bien a la estructura que tiene un EV.

Por otra parte, y dejando a un lado el desarrollo, cabe destacar que se ha implementado un sistema bastante novedoso que, si bien en ciertos aspectos es todavía bastante rudimentario, representa un avance en cuanto a la introducción de nuevas formas de controlar una aplicación, con interfaces que requieran menos atención por parte del usuario, a quien se le debe permitir fijarse más en las tareas que tiene que realizar que en cómo las puede realizar, es decir, que al usuario se le debe posibilitar la interacción con las aplicaciones de manera lo más parecida posible a como hace las cosas en la vida real, sin necesidad de dejarlo restringido a la utilización de un ratón y un teclado.

También se ha trabajado en la implementación de un modelo de comportamiento para los avatares, dotando al EV de mayor credibilidad, lo cual es también un pequeño avance en el campo de los entornos virtuales, ya que el comportamiento que se introduce en los objetos va más allá de la interacción física para meterse en comportamientos más complejos, pues se definen una serie de formas de actuar de los avatares que están basadas en características internas. Si bien es cierto que el diseño de este comportamiento no era uno de los objetivos, sí hay que destacar que el sistema ha servido como campo de pruebas para el modelo de comportamiento.

2. TRABAJO FUTURO

A la vista de las conclusiones obtenidas, el trabajo futuro en lo que se refiere a este tipo de sistemas y al escondite inglés en concreto pasa por dos caminos diferentes.

Por un lado, ha quedado patente que es necesario completar las actuales metodologías de desarrollo para dar soporte a la construcción de sistemas que no se adaptan a las necesidades tradicionales. En concreto, en cuanto a los entornos virtuales, sería deseable la especificación de una forma de construcción de todos los modelos 3D y cómo se relacionan con el resto del diseño. En lo que respecta a las metodologías dirigidas por casos de uso y al método de Larman en concreto, se deben completar con una manera de desarrollar, también de forma sistemática, las funcionalidades del sistema que no dependen de la interacción con el usuario. Para ello, una de las soluciones que desde aquí se quieren proponer es dotar de un mayor peso a los diagramas de transición de estados, especialmente durante la fase de diseño, y una forma de definir las acciones que no realiza el usuario de manera similar a como se hace con los casos de uso. Ya se ha visto una manera de tener en cuenta algunos de estos aspectos, según se ha podido comprobar con la utilización de técnicas propuestas en [Sánchez99], pero todavía queda pendiente la labor de introducirlas y adaptarlas a las metodologías en uso.

Por otra parte, y ya en cuanto al escondite inglés, se proponen varios cursos de acción:

- Estudiar una manera sencilla para que un usuario pueda delegar una función o un conjunto de funciones en su avatar.
- Ampliar el algoritmo de detección de movimiento para que la cámara pueda seguir no sólo el movimiento del usuario por la habitación de juego, sino también otros movimientos, lo que progresivamente vaya haciendo menos necesaria la dependencia de dispositivos de entrada tradicionales, como el ratón y el teclado.
- Desarrollar una forma de comunicación en red fiable, pues en un sistema como éste, que se debe ejecutar en tiempo real, el estado actual de las redes de comunicación suponen un serio obstáculo para su utilización por parte de usuarios que se encuentren muy dispersos geográficamente.

BIBLIOGRAFÍA

[Aviary94] **A Generic Virtual Reality Interface for Real Applications.** Advanced Interfaces Group (AIG) - Universidad de Manchester, Communications Research Group (CRG) - Universidad de Nottingham. http://www.crg.cs.nott.ac.uk/research/systems/AVIARY/ http://aig.cs.man.ac.uk/publications

[Coven99] **Collaborative Virtual Environments**. Universidad de Lancaster. http://coven.lancs.ac.uk

[Deva99] **The DEVA Project.** Advanced Interfaces Group (AIG) – Universidad de Manchester. http://aig.cs.man.ac.uk/systems/deva

[Dive99] **Distributed Interactive Virtual Environment.** Swedish Institute of Computer Science, http://www.sics.se/dive

[Dourish98] **Introduction: The State of Play.** P. Dourish. Computer Supported Cooperative Work: The Journal of Collaborative Computing 7: 1-7. Kluwer Academic Publishers.

[Escape99] **Electronic Landscapes.** Advanced Interfaces Group – Universidad de Manchester, Universidad de Lancaster. http://aig.cs.man.ac.uk/research/escape, http://escape.lancs.ac.uk

[Imbert98] **The Amusement Internal Modelling for Believable Behavior of Avatars in an Intelligent Virtual Environment.** R. Imbert, M. I. Sánchez Segura, A. De Antonio, J. Segovia. Workshop in Intelligent Virtual Environments. ECAI 98 – The 13th Biennial European Conference on Artificial Intelligence. Brighton, UK, 1998.

[Jacobson92] **Object-Oriented Software Engineering: A Use Case Driven Approach.** I. Jacobson. Addison-Wesley, 1992.

[Larman99] **UML y Patrones**. C. Larman. Prentice Hall, 1999.

[Massive95] **Model, Architecture and System for Spatial Interaction in Virtual Environments.** Communications Research Group - Universidad de Nottingham. http://www.crg.cs.nott.ac.uk/research/systems/MASSIVE/

[Maverik00] **The Manchester Virtual Environment Interface Kernel.** Advanced Interfaces Group (AIG) – Universidad de Manchester. http://aig.cs.man.ac.uk/systems/Maverik

[Sánchez99] **Modelado de Entornos Virtuales Basados en la Interacción Social "MEVBIS".** M. I. Sánchez Segura. Tesis de Máster en Ingeniería del Software. Facultad de Informática, UPM, 1999.

[Vilhjálmsson97] **Autonomous Communicative Behaviours in Avataras.** H. Vilhjálmsson. Master of Science Thesis. Massachusetts Institute of Technology.