Desarrollo de Entornos Virtuales para Web

María-Isabel Sánchez-Segura[1], Angélica de Antonio[2], Gonzalo Méndez[2]

[1] Avda. de la Universidad, 30
28911 Leganés (Madrid), España
Tfno: +34 91 624 94 21
Fax: +34 91 624 91 29
Departamento de Informática. Universidad Carlos III de Madrid
misanche@inf.uc3m.es

[2] Campus de Montegancedo s/n
28660 Boadilla del Monte (Madrid), España
Tfno: +34 91 336 69 25
Fax: +34 91 336 69 17
Facultad de Informática. Universidad Politécnica de Madrid
angelica@fi.upm.es, gonzalo@gordini.ls.fi.upm.es

Resumen. Los Entornos Virtuales constituyen un tipo de sistema altamente interactivo para cuyo desarrollo se hace necesario encontrar nuevas técnicas y formalismos especialmente adaptados, ya que los métodos y técnicas clásicos de la Ingeniería del Software han demostrado empíricamente no ser suficientes. En este capítulo, tras una introducción a los entornos virtuales, sus principales características y tipologías, y una revisión de las diferentes tecnologías involucradas en su construcción, se presenta una aproximación al proceso de desarrollo para este tipo de sistemas, el marco metodológico SENDA, profundizando en el conjunto de actividades y técnicas propuestas para la fase de análisis. Por último se hace una revisión de las principales áreas en que estos sistemas están teniendo acogida.

6.1 Introducción

En 1989, Jaron Lanier propone el término de **Realidad Virtual** (RV), y se define como "una simulación interactiva que implica a todos los sentidos, generada por un ordenador, explorable, visualizable y manipulable en tiempo real, dando la sensación de presencia en el entorno". Un aspecto clave en esta definición es que la sensación –visual, auditiva, táctil... – se debe percibir como auténtica por el sujeto.

El término **Entorno Virtual** (EV) fue introducido por investigadores del MIT (*Massachusetts Institute of Technology*) a principios de los años 90 como sinónimo de Realidad Virtual, aunque posteriormente se ha venido utilizando este término para designar algo más general, incluyendo también sistemas en los que la sensación de presencia del usuario en el entorno no es tan fuerte.

En cualquier caso, lo realmente diferente en un sistema de realidad virtual o entorno virtual frente a otros tipos de sistemas interactivos es el hecho de que el usuario pasa a formar parte integrante del entorno, en lugar de interactuar con él desde fuera, y habitualmente adopta algún tipo de representación dentro del entorno que puede manipular en tiempo real. A esta representación de un usuario dentro de un entorno virtual se le ha venido llamando **Avatar**.

Un avatar puede adoptar cualquier forma, desde un simple nombre identificativo, como ocurría en los primeros y más primitivos entornos virtuales, que eran puramente textuales – los conocidos como MUDs (*Multi-User Dungeons*) – hasta un maniquí tridimensional con forma humanoide.

Lo importante de un avatar es que proporciona al usuario una identidad; puede servir a otros usuarios como indicador de presencia, posición y actividad; y es utilizado por el usuario al que representa como medio para interactuar con el resto del entorno.

Los avatares de tipo humanoide facilitan una mayor identificación del usuario con su avatar, pero, dada la mayor complejidad inherente en su construcción y animación, si no se consigue un alto grado de realismo pueden incluso llegar a entorpecer la verosimilitud o credibilidad del entorno, pudiendo ser contraproducentes. En la Figura 6.1 se aprecia el contraste entre un avatar humanoide y otro de diseño fantástico.

Una vez aclarada la terminología más básica en relación con los entornos virtuales, pasaremos a continuación a analizar sus principales características.

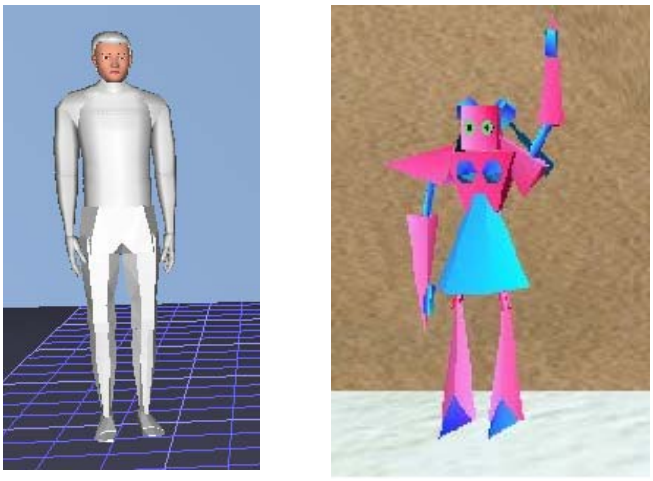


Figura 6.1. Ejemplos de avatares

Este capítulo se estructura del siguiente modo. Este primer apartado de Introducción está dedicado a describir conceptos generales relacionados con los Entornos Virtuales, y especialmente con uno de los elementos principales de estos que son los habitantes de los mismos. En el apartado 6.2 se presentan las distintas tecnologías que se pueden utilizar para desarrollar Entornos Virtuales tanto fuera como dentro de un entorno Web de desarrollo. El apartado 6.3, está dedicado a la descripción detallada del proceso de análisis que se debe llevar a cabo para desarrollar un Entorno Virtual, independientemente de la plataforma e desarrollo que se haya seleccionado. En el apartado 6.4 se recogen las principales aplicaciones de los Entornos Virtuales y por último en el apartado 6.5 se resumen las principales conclusiones que se extraen de la exposición llevada a cabo a lo largo del presente capítulo.

6.1.1 Tipos y características principales de los entornos virtuales

Los entornos virtuales, pueden ser clasificados de acuerdo a múltiples características que serán brevemente analizadas en los siguientes apartados. Es importante identificar el tipo de entorno virtual que se pretende desarrollar de acuerdo con estos parámetros, ya que diferentes variedades de entornos virtuales impondrán requisitos diferentes y exigirán tipos de diseños diferentes.

Dimensionalidad
Quizás la clasificación más inmediata de los entornos virtuales sea de acuerdo con su dimensionalidad en el espacio de presentación, encontrándonos entornos **Textuales** – también llamados Unidimensionales–, **Bidimensionales** y **Tridimensionales**. Los EVs han evolucionado hacia una mayor dimensionalidad en paralelo con la evolución de la tecnología.

Número de usuarios
Una segunda e interesante clasificación de los EVs puede realizarse en función del número de usuarios simultáneos que pueden utilizarlos, encontrándonos EVs **Mono-usuario** y EVs **Multi-usuario**, a los que también se ha dado en llamar **Entornos Virtuales Distribuidos** (DVE, *Distributed Virtual Environments*)

La historia de los DVEs se remonta a la década de los 70 y discurre por dos caminos en paralelo: el mundo de Internet, orientándose fundamentalmente a los juegos en red; y el campo militar, orientándose fundamentalmente a la simulación para el entrenamiento. A estos últimos se les llamó tradicionalmente Simulaciones Interactivas Distribuidas (DIS, *Distributed Interactive Simulation*).

Para facilitar el desarrollo de EVs multi-usuario se han creado diferentes plataformas de desarrollo, como son SPLINE [34], MASSIVE [11], NPSNET [22], o DIVE [7]. Estas plataformas facilitan al desarrollador el manejo de aspectos y problemas específicos que tienen este tipo de sistemas, como son las comunicaciones a través de la red, la sincronización entre las diferentes vistas que los usuarios tienen sobre el entorno, la propagación de eventos de unos usuarios a otros, la escalabilidad del sistema cuando crece el número de usuarios simultáneos, etc.

Grado de inmersión
Un tercer criterio por el que podemos clasificar los entornos virtuales es el grado y cualidad de la sensación de inmersión que ofrecen a sus usuarios. *Grosso modo* se suelen clasificar en **Inmersivos** y **No inmersivos**, llamándose también a estos últimos **Realidad Virtual de Escritorio** (*Desktop Virtual Reality*) por el hecho de que hacen uso de dispositivos de interacción convencionales en un ordenador de escritorio, como son monitor, teclado, ratón o joystick. Los sistemas inmersivos, por el contrario, requieren de dispositivos de interacción muy especiales y generalmente costosos. La inmersión o no-inmersión en un entorno virtual da lugar a dos experiencias fundamentalmente diferentes:

- los sistemas no inmersivos soportan la sensación de "mirar al" EV.
- los sistemas inmersivos soportan la sensación de "estar en" el EV.

Grado de interactividad
Un cuarto criterio de clasificación es el grado de interactividad que ofrecen a sus usuarios. Según este criterio se clasifican en:

- **Pasivos**: son entornos en los que podemos ver, oír, y quizás sentir lo que sucede, pero no es posible controlar lo que ocurre. Corresponde a las llamadas películas dinámicas habituales en parques de atracciones.
- **Exploratorios**: permiten al usuario desplazarse por el entorno virtual para explorarlo. Es el estadio correspondiente a los paseos arquitectónicos y a las obras de arte virtuales.
- **Interactivos**: permiten al usuario explorar y experimentar con el entorno, modificándolo.

Grado de realismo
Si nos centramos en el contenido del EV y su relación con la realidad, podemos hablar de EVs **Realistas**, cuando tratan de reproducir lo más fielmente posible algún espacio o fenómeno real; **Semi-realistas**, cuando básicamente reproducen la realidad, pero adaptándola en algún aspecto para mejorar su comprensión (v.g. podemos adaptar el tamaño en un EV que reproduce el sistema nervioso humano; o la transparencia en un EV que reproduce un motor y deseamos que se vean los componentes internos del mismo; o la escala de tiempo en un EV que nos enseña cómo transcurre la creación de una nueva estrella). Finalmente, podemos encontrarnos EVs totalmente **Fantásticos** y sin relación con la realidad.

Grado de virtualidad
También podemos clasificar los EVs según su grado de virtualidad, pudiendo encontrarnos en cualquier punto de un continuo entre dos extremos constituidos por el mundo real y un mundo totalmente virtual [25]. Aparece así la llamada **Realidad Mixta**, incluyendo sistemas de **Realidad Aumentada**, cuando es el mundo real el que se ve complementado con objetos virtuales, y sistemas de **Virtualidad Aumentada**, cuando objetos del mundo real pueblan el mundo virtual.

6.1.2 Habitantes en un entorno virtual

En un entorno virtual podemos encontrarnos dos tipos diferentes de habitantes:

- **Avatares** puros, que representan y son controlados por un usuario.
- **Agentes virtuales** autónomos, que no son controlados por un usuario sino por el propio sistema.

Más precisamente podemos definir un agente virtual como un agente software, más o menos inteligente, que está encarnado (es decir, adopta una representación corpórea, generalmente con forma humana, a la que se suele llamar *embodiment*), y que habita un entorno virtual. El hecho de que habite un EV significa que no sólo percibe observaciones de un entorno externo, sino que forma parte del mismo entorno, y debe ser capaz de desenvolverse, percibir e interactuar en él.

La información a percibir no es una representación simbólica abstracta especialmente adaptada, sino información compleja de tipo visual, auditivo, táctil, etc.

A diferencia de los agentes software inteligentes no encarnados, los humanos observarán no sólo el resultado de sus acciones sobre el entorno, sino la forma en que se producen dichas acciones, y una parte importante de sus acciones implicarán la manipulación de su propio cuerpo. En estas acciones su comportamiento debe ser verosímil para un observador humano.

El diseño de avatares y agentes virtuales comparte la necesidad de dotar a ambos de un cuerpo y ciertas posibilidades de acción, pero un agente virtual requiere además el diseño de sus capacidades perceptivas y sus capacidades de razonamiento y toma de decisiones, aspectos estos totalmente ausentes en el diseño de un avatar.

En la Figura 6.2 se pueden ver los tres componentes de un agente virtual, según un modelo general de organismo.

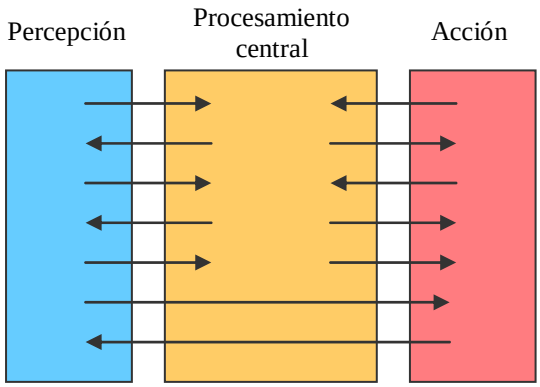


Figura 6.2. Modelo general de organismo de un agente virtual

6.2 Tecnologías para el desarrollo de entornos virtuales dentro y fuera de la web

La construcción de un entorno virtual tridimensional (3D) implica la utilización de diversas herramientas para completar todo el proceso de desarrollo. Estas herramientas, por un lado, permiten la creación de los elementos tridimensionales que estarán presentes dentro del EV, como pueden ser las estancias que conforman el mundo virtual, los avatares que representan a los usuarios y demás habitantes del mundo, y los objetos con los que es posible interactuar. Por otra parte, será necesario otro tipo de herramientas que permitan, una vez se dispone de los modelos 3D, dotarles de comportamiento y posibilitar la interacción con ellos.

Como sucede con los entornos de desarrollo tradicionales, cada vez es más habitual encontrar herramientas que incorporen todas las funcionalidades necesarias para desarrollar un EV, desde la creación de los modelos 3D hasta la interacción con dispositivos de realidad virtual, pasando por mecanismos de animación y lenguajes de programación que permiten dotar de comportamiento a los elementos presentes en el EV.

A lo largo de los siguientes apartados se van a describir distintas herramientas que se utilizan para realizar las tareas descritas, aunque no se pretende realizar un estudio detallado, pues se encuentra fuera de los objetivos del presente capítulo.

6.2.1 Herramientas para la elaboración de modelos 3D

Los objetos tridimensionales que se utilizan para construir los EVs deben cumplir unas características muy concretas, especialmente debido a la capacidad de procesamiento que exige la representación de un entorno virtual. De esta manera, aunque en la actualidad existen herramientas de modelado que permiten realizar trabajos de enorme complejidad (v.g. mediante el uso de NURBs (*NonUniform Rational B-splines*), distintos tipos de iluminación, sistemas de partículas), su potencia sólo se aprovecha para labores de cinematografía.

En lo que a la construcción de EVs se refiere, lo que se le exige a estas herramientas es poder crear los objetos 3D haciendo uso de polígonos, generalmente triángulos, colorearlos, mejorar su aspecto con el uso de texturas y, en caso necesario, construir animaciones para mostrar el comportamiento de los objetos dentro del entorno virtual. Adicionalmente, es deseable que estas herramientas guarden los objetos 3D en algún formato estándar que sea fácilmente accesible para otras herramientas, ya que, aunque es posible emplear utilidades que conviertan un formato gráfico en otro, los resultados no son siempre los deseados.

Una de las herramientas más potentes y populares es 3D Studio, que proporciona todas las funcionalidades mencionadas anteriormente. Esta herramienta es especialmente destacable porque ha dado lugar a un estándar de facto para el formato de los ficheros donde se guardan los objetos 3D, el formato 3ds, que es leído y generado por multitud de aplicaciones. Como característica adicional, también proporciona un pequeño lenguaje de *scripts* que, entre otras cosas, permite la construcción de exportadores que conviertan los objetos creados a formatos propietarios que resulten de mayor utilidad a la hora de construir el EV.

Los modelos generados por estas herramientas serán posteriormente importados desde otras aplicaciones para construir el EV propiamente dicho.

6.2.2 Herramientas gráficas

Debido sobre todo a su carácter visual, estas herramientas permiten realizar un desarrollo bastante rápido de los EVs, especialmente si lo comparamos con el tiempo de desarrollo que requiere el uso de librerías. Como contrapartida, en general constituyen un método bastante menos potente para construir los EVs, ya que no permiten tocar aspectos de bajo nivel del motor gráfico, cambiar el modo de ejecución y, en general, hacer algo que no proporcione la herramienta en sí.

WorldUp

WorldUp es una herramienta de autor para construir Entornos Virtuales. Esta herramienta permite crear los objetos 3D que van a formar el EV, y también da la posibilidad de importar los objetos diseñados con otras aplicaciones siempre que se utilicen formatos como 3ds o VRML. En el caso de crear los objetos con WorldUp, se pueden realizar tareas comunes como rotarlos, cambiarles el tamaño, el color o la textura.

Una vez cargados los objetos 3D, permite manejarlos como si se tratasen de objetos de un lenguaje de programación orientado a objetos, de manera que permite crear clases de objetos, subclases de las anteriores que heredan sus propiedades, definir atributos o asignar comportamientos como si de métodos se tratase, aunque, en este último caso, los *scripts* que contienen los comportamientos se le asignan a los objetos individualmente.

WorldUp proporciona de serie una lista de dispositivos que se pueden utilizar para navegar por los entornos virtuales creados con esta herramienta, pero también admite la utilización de módulos externos que permitan el uso de dispositivos no contemplados por WorldUp.

La manera habitual de ver un escenario es utilizar un reproductor para WorldUp. El EV se guarda en un único archivo que es abierto por el reproductor. Los reproductores pueden ser una aplicación independiente o estar contenidos en otra aplicación. También puede utilizarse un *plug-in* para los navegadores web, o un control Active-X para integrarlo con aplicaciones desarrolladas en C++ o Visual Basic.

Active Worlds

Active Worlds es una aplicación multiusuario que permite a los usuarios conectarse a una serie de entornos virtuales en 3D y conversar o comprar productos en ellos. Active Worlds proporciona servidores de mundos 3D construidos con objetos con el formato RWX (*RenderWare Script format*), formato que puede ser utilizado por varias librerías gráficas, como DirectX, y también es el formato que siguen los objetos de algunos juegos desarrollados para las videoconsolas Xbox, GameCube y Playstation-2. En estos servidores de mundos se instala un software que permite ejecutar a la vez diferentes entornos virtuales. Una vez que el EV está activo en un servidor, un usuario puede conectarse a éste por medio de un navegador y visualizar el mundo al que se ha conectado. Algunos de los EVs más grandes creados hasta la fecha tienen registrados más de 200.000 ciudadanos.

Para visualizar el EV no es necesario cargar todos sus objetos ni todos los avatares que representan al resto de los usuarios conectados, sino que simplemente se visualizan aquellos que el usuario puede ver en ese momento. Cuando se visualiza un objeto por primera vez, la información referente a él se descarga en la máquina del usuario y se almacena en ella, de manera que no es necesario volver a descargarla en posteriores ocasiones.

Uno de los aspectos más importantes de Active Worlds, y que lo diferencia de otras aplicaciones que proporcionan accesos a entornos virtuales, es la capacidad que tienen los usuarios de construir dentro del EV. Esto quiere decir que un usuario puede, en una zona vacía de un EV, añadir diferentes objetos y edificios. Para que un usuario construya en un EV debe clonar un objeto preexistente y situar el clon en un lugar que no esté ocupado. Posteriormente puede cambiar las características de este nuevo objeto. Además de construir en el EV, un usuario puede construir también el propio avatar que lo represente, en vez de elegir uno predeterminado.

Los objetos del EV deben tener el formato RWX, en el que se especifican las características del objeto. Existen diferentes formas de crear un objeto de este tipo. Una de ellas es editar el *script* del objeto y ver cómo ha quedado en un visor de objetos de este formato. Otra manera es utilizar una herramienta de modelado visual de objetos.

Por otra parte, para programar aplicaciones y hacer que utilicen Active Worlds, existen varios SDK (*Software Development Kit*) y una API (*Application Programming Interface*). La mayoría de los métodos de la API están orientados a la conexión con el servidor, la validación del usuario que se conecta al EV, a sus capacidades de teletransporte (dar un salto en el EV y aparecer en otro lugar), capacidad de volar, etc.

Active Worlds está pensado para interrelacionarse con otros usuarios y charlar, es decir, el EV es un medio y no un fin, por lo que no existen funciones complejas para la manipulación de objetos.

Jack

Es una aplicación desarrollada por la Universidad de Pennsylvania que está orientada al estudio de la ergonomía del ser humano en diferentes lugares de trabajo. Su principal valor radica en la capacidad para la simulación de cuerpos humanos con sus características biomecánicas, antropométricas y ergonómicas.

Entre las características principales de Jack se incluye un sistema para modelar figuras humanas articuladas con detección de colisiones y cinemática inversa. Estas figuras se encuentran en una librería que incorpora la propia aplicación, la cual nos permite cambiar las características de cualquiera de estas figuras para adaptarlas a nuestras necesidades. Las figuras pueden tener definidas restricciones ergonómicas, y el sistema de simulación tratará de cumplirlas con la mayor exactitud.

La aplicación también permite crear objetos simples e importar objetos creados con otras herramientas, de manera que se pueden crear escenarios complejos con bastante facilidad.

Otro punto destacable es la incorporación de los lenguajes de programación Tcl, Python y Lisp, por medio de los cuales es posible crear un entorno virtual con nuevas ventanas y controles y con las interacciones y comportamientos de los objetos que defina el desarrollador del EV.

6.2.3 Librerías de programación

Las librerías de desarrollo de gráficos tridimensionales son una potente herramienta sobre la que se apoyan las aplicaciones descritas en el apartado anterior. Aunque existe una amplia variedad de librerías, la mayoría de los EVs se desarrollan utilizando una de las dos que se van a describir a continuación: DirectX-Direct3D y OpenGL.

DirectX – Direct 3D

DirectX es una API desarrollada por Microsoft que tiene una amplia difusión para el desarrollo de juegos, aunque, cada vez más, se utiliza también para otro tipo de aplicaciones, y en concreto para el desarrollo de EVs.

Esta librería se encuentra disponible únicamente para la plataforma Windows, lo cual supone una gran desventaja frente a OpenGL, pero, a diferencia de ésta, DirectX se compone de una serie de módulos que proporcionan una funcionalidad mucho más amplia que la que provee OpenGL.

El más relacionado con el tema que nos ocupa es DirectX Graphics, que proporciona manejo de gráficos 2D (DirectDraw) y 3D (Direct3D) de manera similar a OpenGL, aunque también incluye funciones de más alto nivel para facilitar la construcción y manejo de los entornos tridimensionales.

Direct3D maneja su propio formato para los objetos 3D, el formato *X*, lo cual no impide utilizar otros formatos gráficos, eso sí, sin aprovechar las facilidades que da Direct3D para operar con el formato X. Este formato almacena la descripción geométrica de los objetos, su relación jerárquica e incluso las animaciones que se puedan haber diseñado con herramientas de modelado. Además, el SDK de DirectX incorpora *plug-ins* para 3DStudio y Maya que permiten que estas herramientas generen ya los modelos en formato *X*.

Adicionalmente, DirectX incluye otros módulos, como son DirectInput, para el manejo de distintos elementos de interacción como teclado, ratón o joystick, DirectSound, para la reproducción de música y efectos sonoros, o DirectPlay, para la gestión de conexiones de red en aplicaciones multiusuario.

OpenGL

OpenGL es una librería que sirve para el desarrollo de gráficos 2D y 3D. Creada inicialmente por Silicon Graphics en 1992, actualmente se encarga de su mantenimiento y desarrollo el *OpenGL Architecture Review Board*, un consorcio que cuenta con un amplio respaldo en el mundo de la industria.

Una de las razones por las que su uso es tan extendido se encuentra en el hecho de que es una librería que se encuentra disponible en una amplia gama de plataformas, como Windows, Linux o Irix, de manera que el código desarrollado es fácilmente portable entre ellas. Además, se puede utilizar con distintos lenguajes de programación, como C, C++, Ada, Fortran, Perl y Java, lo cual ofrece grandes atractivos a una gran variedad de desarrolladores de software.

OpenGL no proporciona funciones de alto nivel para describir modelos de objetos tridimensionales, sino que se deben construir los modelos deseados a partir de un pequeño conjunto de primitivas geométricas, como puntos, líneas y polígonos. La librería de utilidades de OpenGL (GLU, *OpenGL Utility Library*) proporciona más funcionalidades para el modelado, como superficies cuádricas y curvas y superficies NURBS. GLU es estándar en todas las implementaciones de OpenGL.

A través del trabajo con vectores y matrices, OpenGL permite realizar transformaciones de objetos, como traslaciones, rotaciones o cambios de escala. Además, también posibilita el manejo de luces, colores o cámaras, de manera que se pueden controlar, a muy bajo nivel, todos los aspectos que se desee en cuanto a la representación del entorno virtual. Gracias a su fácil integración con distintos lenguajes de programación es posible dotar a los objetos del EV de comportamientos tan complejos como se pueda imaginar.

Es importante destacar que OpenGL sirve exclusivamente para la construcción de los entornos virtuales, no ofreciendo la posibilidad de crear y manejar ventanas, sonidos, conexión entre distintos clientes, etc., tareas que deberán ser realizadas mediante otros mecanismos que facilite el lenguaje de programación utilizado o mediante librerías adicionales.

Sobre OpenGL se han construido extensiones que proporcionan funciones de más alto nivel que permiten un uso más sencillo de esta librería sin necesidad de implicar al desarrollador en el manejo de detalles de bajo nivel que no le interesan. De especial interés resulta OpenGL Performer, una utilidad desarrollada específicamente para facilitar el desarrollo de simulaciones y aplicaciones de Realidad Virtual.

6.2.4 Desarrollo para web

Como casi cualquier otro tipo de aplicación actual, los EVs no han podido escapar del atractivo que ofrece Internet para su desarrollo y distribución, ya que la posibilidad de crear un EV que sea accesible a través de un navegador web hace que los potenciales usuarios del EV crezcan de manera exponencial. Para ello, basta añadir uno de los numerosos *plug-ins* existentes para los distintos navegadores que son capaces de visualizar el EV dentro de la ventana del navegador, como pueden ser:

- Cosmo Player: uno de los plug-ins más antiguos para visualización de VRML, aunque estuvo disponible para diversas plataformas, actualmente sólo se desarrolla para el sistema operativo IRIX, de Silicon Graphics.
- Cortona: ofrece soporte para VRML y Macromedia Flash, y está optimizado para su ejecución en PCs actuales. Se integra, además de con navegadores, con herramientas ofimáticas.
- Octaga: soporta VRML y X3D. En general, ha demostrado ser más rápido que el resto de plug-ins.
- blaxxun Contact: soporta VRML y X3D, y puede ser integrado fácilmente con otras aplicaciones.

Una de las mayores desventajas de estas aplicaciones se encuentra en su lentitud, pues una vez que se accede a Internet la velocidad de las comunicaciones es mucho menor que dentro de una red local. Cada vez que el usuario se conecta a un EV es necesario que éste se descargue en su máquina, lo cual suele llevar bastante tiempo. Si además el EV es multiusuario, la comunicación con el resto de usuarios se realizará con una velocidad bastante baja, motivo por el cual las aplicaciones que se realicen no podrán ser excesivamente dependientes de la velocidad de las comunicaciones.

VRML

VRML (*Virtual Reality Modeling Language*) es un lenguaje para describir simulaciones interactivas multiusuario. Su origen data del año 1995, cuando se publicó la primera especificación para VRML 1.0, cuyas capacidades de interacción quedaban bastante limitadas.

La especificación de VRML más actual es VRML97 [14], [15]. Aunque VRML está siendo substituido paulatinamente por X3D, su amplia utilización hace aún que sea interesante dedicarle alguna atención.

Como se define en el mencionado estándar, VRML es un formato de fichero para describir objetos y mundos tridimensionales interactivos. Por estar basado en la web, VRML permite que los objetos sean hiperenlaces a otros EVs, pero también a páginas web, imágenes, vídeos y otros tipos de documentos.

Los ficheros de VRML son archivos de texto que describen la geometría de los objetos 3D, su jerarquía y, si se desea, pueden incluir *scripts* que definan el comportamiento de los objetos. Sin embargo, estos comportamientos son de un tipo bastante elemental, y las posibilidades de interacción tampoco son demasiado sofisticadas, por lo que para sacarle todo el partido posible es necesario combinarlo con algún lenguaje de programación, generalmente Java.

La combinación de VRML y Java, ambos disponibles para una amplia variedad de plataformas, hace posible la construcción de EVs de una complejidad similar a la de otros construidos con otros lenguajes (v.g. C++ y OpenGL) con las mencionadas ventajas y desventajas de ejecutarse en un entorno web.

X3D

X3D (*eXtensible 3D*) es la evolución de VRML hacia un lenguaje de especificación de entornos tridimensionales [16].

El objetivo ha sido definir un nuevo lenguaje que cumpla una serie de requisitos que VRML no cumple, como separar la arquitectura de la codificación de los datos, añadir nuevos objetos y comportamientos o proporcionar distintas APIs para el manejo de los EVs.

En la actual especificación se contempla que X3D puede manejar gráficos 2D y 3D, animaciones, sonido y vídeo. También se da mayor soporte al manejo de dispositivos de interacción, control de la cámara dentro de la escena, mecanismos de detección de colisiones e integración con distintos lenguajes de programación (concretamente, ECMAScript y Java) y otros estándares como H-Anim o el protocolo DIS (*Distributed Interactive Simulation*).

Para mantener la compatibilidad con VRML se ha tomado la decisión de que los ficheros X3D se puedan definir de dos maneras. Una de ellas es respetando las sintaxis de VRML, mientras que la otra es a través del uso de una especificación en lenguaje XML.

Como se puede apreciar, por tanto, X3D continúa el camino emprendido por VRML, incorporando nuevos mecanismos para hacerlo más abierto, flexible y ampliable, pero manteniendo en esencia todas las características de VRML.

H-Anim

H-Anim es una especificación que originalmente surge para definir cómo construir figuras humanas con VRML [17]. Actualmente, con el desarrollo de X3D, se está definiendo un nuevo estándar que, entre otras características, presenta unas guías sobre cómo elaborar estas figuras con VRML y con X3D.

La necesidad de H-Anim surge del hecho de que existían diversas aplicaciones que servían para diseñar y animar figuras humanas. Aunque cada una realizaba su cometido con bastante éxito, la diferente representación de las figuras hacía bastante difícil la conversión de formatos entre distintas aplicaciones. Esto era especialmente notable al utilizar herramientas de modelado de un fabricante y aplicaciones de captura de movimientos de otro distinto. Con la adopción de H-Anim se facilita la tarea de integrar el resultado de ambos sistemas (v.g. para elaborar un avatar con movimientos similares a los humanos).

Algunos tipos de animaciones humanas no dependen de las dimensiones del cuerpo humano, mientras que otras sí. Teniendo esto en cuenta, es posible compartir las animaciones que no dependen de las dimensiones del cuerpo, mientras que si se mantienen las proporciones, también será posible adaptar las animaciones que sí dependen de las dimensiones del cuerpo humano.

Para tener todo esto en cuenta, H-Anim sugiere las dimensiones y las posiciones que deben tener todas las partes del cuerpo. Como además es habitual que en muchas aplicaciones no se requiera representar el cuerpo humano con el mismo grado de detalle, en el estándar H-Anim las articulaciones se han organizado en cuatro niveles de detalle, de manera que una animación hecha para una figura que se ajusta a un nivel de detalle será utilizable con figuras del mismo nivel de detalle o mayor.

En total, en la especificación actual de H-Anim se definen 76 características significativas en la anatomía del cuerpo humano, aunque tras su posterior traducción a articulaciones y objetos se pueden contabilizar hasta 89 elementos, por replicación de algunos puntos significativos.

6.2.4 Plataformas de desarrollo de EVs

Uno de los principales intereses en la construcción de EVs lo constituye la posibilidad de construir mundos que den soporte a múltiples usuarios de manera distribuida, y que estos mundos que se construyen sean fácilmente escalables.

A continuación se describen una serie de proyectos que han servido para dar un gran impulso al desarrollo de EVs con las mencionadas características.

Dive

DIVE (*Distributed Interactive Virtual Environment*) [8] es una herramienta que está orientada al desarrollo de EVs colaborativos de gran tamaño. Está especialmente pensada para soportar la comunicación de diversos clientes que se comunican a través de una red. Esta herramienta se ha desarrollado desde el año 1991, y actualmente se pueden encontrar versiones que se ejecutan sobre Windows, Linux, Solaris, Irix y HPUX.

DIVE es capaz de leer distintos formatos de objetos 3D, entre ellos VRML, y posteriormente el comportamiento de estos objetos se puede programar a través del uso de *scripts* escritos en Tcl. Estos *scripts* se pueden ejecutar como consecuencia de distintos eventos del sistema, como interacciones o colisiones, lo cual posibilita un comportamiento bastante complejo.

La comunicación está basada en un sistema P2P, de manera que no es necesario contar con un servidor de mundos, y toda la comunicación entre clientes se realiza mediante un proceso de multienvío y una base de datos distribuida del mundo. Esta base de datos se mantiene en cada cliente, de manera que la información del mundo perdura mientras al menos un cliente se encuentre activo.

Para que todo este volumen de comunicación funcione en tiempo real, y para que lo haga con redes cuyo ancho de banda no es muy grande, como sucede con Internet, se ha optado por la solución de no mantener una consistencia total entre las vistas del mundo en cada cliente, y se han implementado servicios que se aseguren de mantener un alto grado de consistencia, aunque ésta no sea total.

Existen varias opciones para utilizar DIVE en la construcción de un entorno virtual, entre las que se encuentran: codificar el EV en C/C++ o Java y compilarlo con las librerías de DIVE, usar DIVE como un *plug-in*, comunicarse con DIVE por TCP/IP o, como parecen preferir los autores, a través de *scripts* escritos en Tcl/Tk.

Massive

Massive es una serie de proyectos cuyo propósito es, en esencia, dar soporte al desarrollo de entornos virtuales multiusuario.

La primera versión de este proyecto [11] consistía en una aplicación de teleconferencia a la que se le dio la apariencia de un EV, y que era capaz de soportar alrededor de 10 usuarios, los cuales podían comunicarse con una mezcla de texto, sonido y el entorno 3D.

La segunda versión, Massive-2, también conocida como CVE (*CRG Virtual Environment*) [3], amplía y mejora las capacidades de la versión anterior, especialmente en lo que se refiere al modelo de interacción espacial. Para esta versión se desarrolló CRGShare, unas librerías escritas en C y diseñadas para ayudar a los desarrolladores que quieran implementar EVs distribuidos basados en Massive-2. Están estructuradas en forma de componentes, de manera que los desarrolladores puedan elegir los componentes que les convengan para su EV.

La actual versión, Massive-3, conocida como HIVEK (*HIVE Project Kernel*) [12], es un sistema de Realidad Virtual multiusuario y distribuido en el que se ha puesto un mayor énfasis en la escalabilidad. Al igual que sus predecesores, soporta comunicación con gráficos 3D y sonido en tiempo real, pero en este caso se ha eliminado el texto. Está desarrollado usando parte de las librerías que componen CRGShare, y gran parte de la arquitectura está basada en agentes.

Spline

Spline (*Scalable Platform for Large Interactive Networked Environments*) [34] es una herramienta cuya principal característica es ofrecer interfaces abiertas que faciliten la interoperabilidad entre EVs desarrollados de manera independiente.

Una de las características a destacar de esta herramienta es la definición del concepto de **locale** como forma de dividir un entorno en subentornos que se gestionan por separado, concepto que ha sido utilizado posteriormente en otros sistemas (v.g. Massive). Gracias a esta forma de construir los EVs, Spline hace posible que los usuarios puedan hacer cambios temporales o permanentes en el EV mientras éste se está ejecutando, de manera que puede evolucionar según los deseos de los usuarios.

La comunicación entre distintos usuarios y aplicaciones que hacen uso de un EV construido con Spline se hace a través de modificaciones en un modelo compartido del mundo virtual, donde se sabe la posición de cada objeto, su aspecto, qué acciones está realizando y qué sonidos está reproduciendo, y este modelo se reproduce en todos los clientes para mantener la eficiencia y la escalabilidad. El hecho de que el mundo esté dividido en *locales* hace que la información que hay replicar no sea excesivamente grande. Cuando en un cliente se realiza algún cambio, se comunica al resto a través de un proceso de multienvío.

Spline se ha utilizado para construir, como mundo más destacado, Diamond Park [34], un EV que representa una feria universal con edificios que recuerdan diversas culturas y periodos históricos y donde se pueden realizar actividades como diseñar paisajes o montar en bicicleta.

6.3 ¿Cómo abordar el desarrollo de un entorno virtual?

Hasta aquí se ha introducido al lector en el concepto de EV y los elementos que lo rodean; además se han revisado las principales herramientas que asisten al desarrollador en el proceso de implementación. El interés de este apartado es conciliar el desarrollo de este tipo de sistemas con la disciplina de la Ingeniería del Software, que tiene mucho que decir en cuanto a la manera óptima de plantear el desarrollo de estos sistemas para asegurar la calidad de los mismos.

En el caso de los desarrollos orientados a la construcción de EVs, el hecho de aplicar los procesos de desarrollo tal cual los proponen las diferentes metodologías existentes presenta problemas en forma de carencias en la manera de definir los requisitos, en la descripción visual de los EVs, y por lo tanto, en el diseño de éstos, problemas a la hora de gestionar los proyectos, sobre todo en la tarea de estimación, así como problemas de verificación del trabajo que debía realizarse, imposibilidad de reutilizar, etc. Con el fin de adaptar los modelos de proceso convencionales al desarrollo de un EV se describió el Marco Metodológico SENDA [29].

A lo largo de esta sección se describirá en primer lugar una visión global de procesos y su justificación dentro de SENDA; a continuación se describe en profundidad el proceso de Análisis, incluyendo ejemplos de aplicaciones en las que se ha llevado a cabo el proceso de Análisis propuesto en SENDA. Dado que este capítulo se centra en el proceso de análisis, si el lector desea conocer más detalles sobre el resto de tareas que propone SENDA para el resto de procesos de desarrollo puede consultar [30]. Las dos aplicaciones concretas sobre las que se basan los ejemplos que se incluyen a lo largo de esta sección son PRVIR, aplicación para el entrenamiento en centrales nucleares; y Escondite Inglés, para el entretenimiento a través de Internet.

6.3.1 Visión general del proceso

Debido a las características especiales de estos desarrollos, los procesos de Diseño e Implementación, se van a ver descompuestos como se indica en la Figura 6.3. El proceso de Diseño se descompone en los siguientes procesos: **Diseño 3D del EV**, **Diseño de Elementos Multimedia**, **Diseño de la Arquitectura Interna de los Componentes** y **Diseño del Sistema**. El proceso de Implementación se divide en dos procesos: **Implementación de Componentes de Soporte** e **Implementación del Módulo Principal**. El motivo de esta división en procesos se debe al tipo de tareas que en cada uno de ellos se realiza y a la relación existente entre procesos del modelo global. Gracias a esta descomposición se consigue mejor seguimiento y visibilidad sobre los desarrollos. No en todos los procesos son nuevas todas las tareas ni todas las técnicas propuestas; por este motivo se ha utilizado una notación para diferenciar los procesos en los que todas las tareas y técnicas propuestas son nuevas, los procesos en los que sólo algunas de las tareas o técnicas lo son, y aquellos en los que las tareas implican el uso de técnicas previamente descritas en otras disciplinas.

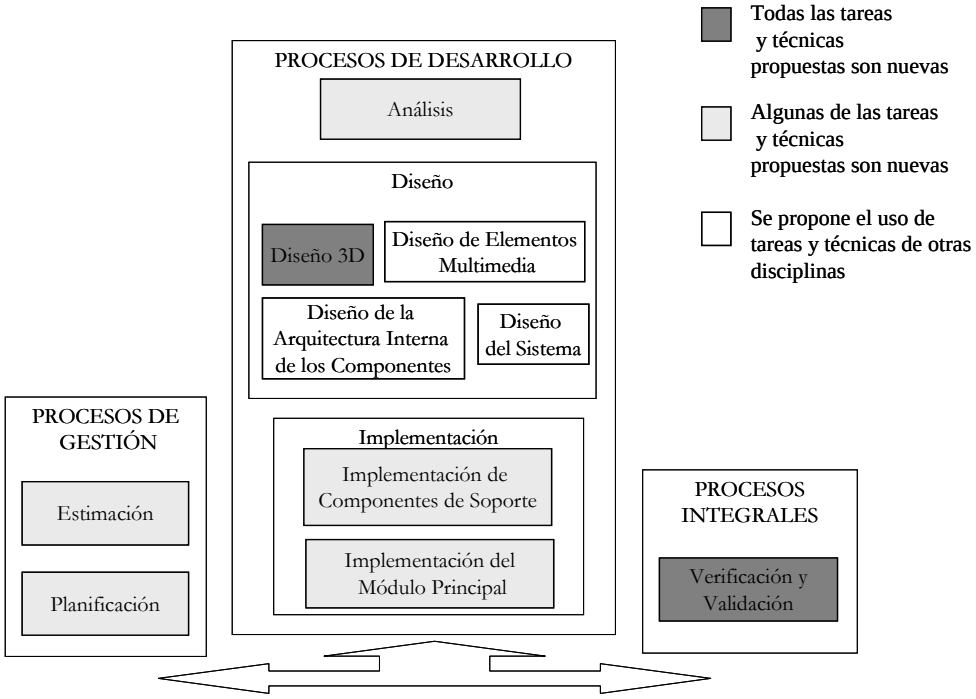


Figura 6.3. Modelo de Procesos Propuesto Detallado

Para la representación gráfica de las tareas y sus relaciones dentro de los procesos y con tareas de otros procesos se utilizará la notación de símbolos descrita en [20] para modelado de procesos. Para nombrar las tareas se utilizará una notación significativa compuesta por Acrónimo del Proceso + Acrónimo de la Tarea. Los acrónimos para los procesos son los siguientes:

- A: Proceso de Análisis
- D3D: Proceso de Diseño 3D
- DEM: Proceso de Diseño de Elementos Multimedia
- DS: Proceso de Diseño del Sistema
- DAI: Proceso de Diseño de la Arquitectura Interna de los Componentes
- ICS: Proceso de Implementación de los Componentes de Soporte.
- IMP: Proceso de Implementación del Módulo Principal
- E: Proceso de Estimación del Proyecto.
- P: Proceso de Planificación del Proyecto.
- V&V: Proceso de Verificación y Validación.

En la Figura 6.4 se muestran los procesos de la Figura 6.3 en más detalle, indicando las tareas que componen cada proceso así como la relación entre tareas dentro de cada proceso. La relación entre tareas de procesos distintos se describe en las distintas secciones dedicadas a cada proceso.

En todo sistema software que se quiera desarrollar, el primer paso en el desarrollo debe ser la extracción de requisitos. Concretamente en los EVs, debido a la evolución constante que están sufriendo, es preciso tomar una serie de decisiones tecnológicas que van a marcar el resto del desarrollo. Lo antes posible se debe decidir el tipo de dispositivos de realidad virtual, el software de desarrollo, el hardware, etc., con el fin de comprobar la compatibilidad de dichos elementos. Por esto se ha creado la tarea de **Definición de Requisitos Específicos**, dentro del proceso de Análisis. Así se puede reforzar la toma de dichas decisiones, que son específicas para EVs.

Además, como en todos los sistemas software, se debe hacer la extracción de requisitos funcionales. Una vez que se tenga la lista de requisitos del sistema, y tomando como axioma que la orientación a objetos es la que mejor se adapta a los EVs, se deben elaborar los casos de uso para todos aquellos requisitos que impliquen una interacción del usuario con el sistema, tal como se propone en [18]. Pero los casos de uso se quedan cortos al intentar representar determinados requisitos asociados a funcionalidad que no será demandada directamente por el usuario cuando haga uso del EV. Para resolver esta carencia, se propone una nueva técnica, los **conceptos de uso**, que resolverán el problema planteado por los servicios del EV que no son demandados directamente por el usuario.

Tanto los casos como los conceptos de uso se catalogarán basándonos en un conjunto de categorías que se proponen en el proceso de análisis. Dichas categorías vienen a agrupar las funcionalidades del sistema, basándose en los distintos modos de interacción identificados en la taxonomía de componentes propuesta en SENDA [31]. Así, se tendrán casos o conceptos de uso de percepción, animación, razonamiento, decisión, etc.

Además de los requisitos específicos y funcionales, existe otro conjunto de requisitos asociados al aspecto del EV. Dichos requisitos, como no son relativos a la funcionalidad del sistema, suelen quedar sin describir, al no tener cabida en el análisis de requisitos clásico. Para poder contemplarlos adecuadamente, se ha propuesto un proceso específico ubicado dentro de los de diseño, llamado **proceso de Diseño 3D**. Este proceso es necesario para poder alcanzar dos objetivos:

1. Extraer información sobre el aspecto que debe tener el EV
2. Poder comunicarse con el modelador, en un lenguaje común, que permita al diseñador del sistema y al modelador entenderse. En estos casos el lenguaje natural falla, debido a que la formación del diseñador de sistema (informático) y el modelador (experto en artes gráficas) es muy diferente.

Lo mismo ocurre con los elementos multimedia. Aunque son requisitos que tienen menos riesgo que los del modelado 3D del EV, también son requisitos de aspecto importantes. Por eso se ha creado el **proceso de Diseño de Elementos Multimedia.**

Una vez que se han extraído los requisitos de aspecto, se debe comprobar que dicha información especificada es la correcta. Para esto, se deben construir maquetas, vídeos, etc. Esto se ha incluido como parte del proceso de diseño del sistema, concretamente de la tarea de Diseño de la Interfaz. En dicha tarea se aprovechará para mostrar al usuario ejemplos de distintas alternativas de navegación por el EV. Esto es algo común a muchos sistemas software, pero es especialmente relevante en los EVs. Como bien se apuntaba en [5], la diferencia esencial entre la interfaz de un sistema tradicional y un EV es que, en el primero la interfaz sirve para ofrecer funcionalidad al usuario, mientras que en el segundo sirve además para conseguir que el usuario se sienta parte del EV. Las técnicas de diseño participativo o centradas en el usuario, son útiles para llevar a cabo esta tarea.

A partir del documento de conceptualización, en el que aparecerán todos los requisitos funcionales del EV, se extraerán clases para construir el modelo de estructura estática. Dicho modelo incluirá, además, clases provenientes del proceso de Diseño 3D y del Diseño de Elementos Multimedia. Tanto las clases como los métodos se obtienen de cada caso y concepto de uso, del proceso de Diseño 3D y del proceso de Diseño de Elementos Multimedia. Como los casos y conceptos de uso se han catalogado en categorías, será fácil saber de qué tipo de categoría es cada método. Es decir, si un caso de uso está catalogado dentro de la categoría de percepción, entonces deberá estar en alguna clase como método de detección, ya que es preciso dotar a la clase del correspondiente método de detección de modo que algo pueda ser percibido.

Dentro de las categorías de Casos y Conceptos de Uso, son especialmente críticas las que se refieren a lo que los componentes del EV pueden percibir, las que representan características internas, como personalidad, humor, etc., las que se refieren al modo en que razonarán y las que representan las reacciones posibles ante una interacción. Dichas reacciones pueden suponer desde una simple modificación en una variable hasta la representación de una escena muy compleja en el EV.

Para poder diseñar en detalle todo lo anterior, se propone un proceso de **Diseño de la Arquitectura Interna de los Componentes**. En dicho proceso se diseñará:

1. Cómo deben detectar o percibir los componentes del EV.
2. Cómo afectarán las características internas de los componentes al comportamiento de estos.
3. Cómo funcionarán los mecanismos de razonamiento partiendo de la detección de una interacción.
4. Cómo se representarán externamente las acciones de los componentes del EV.

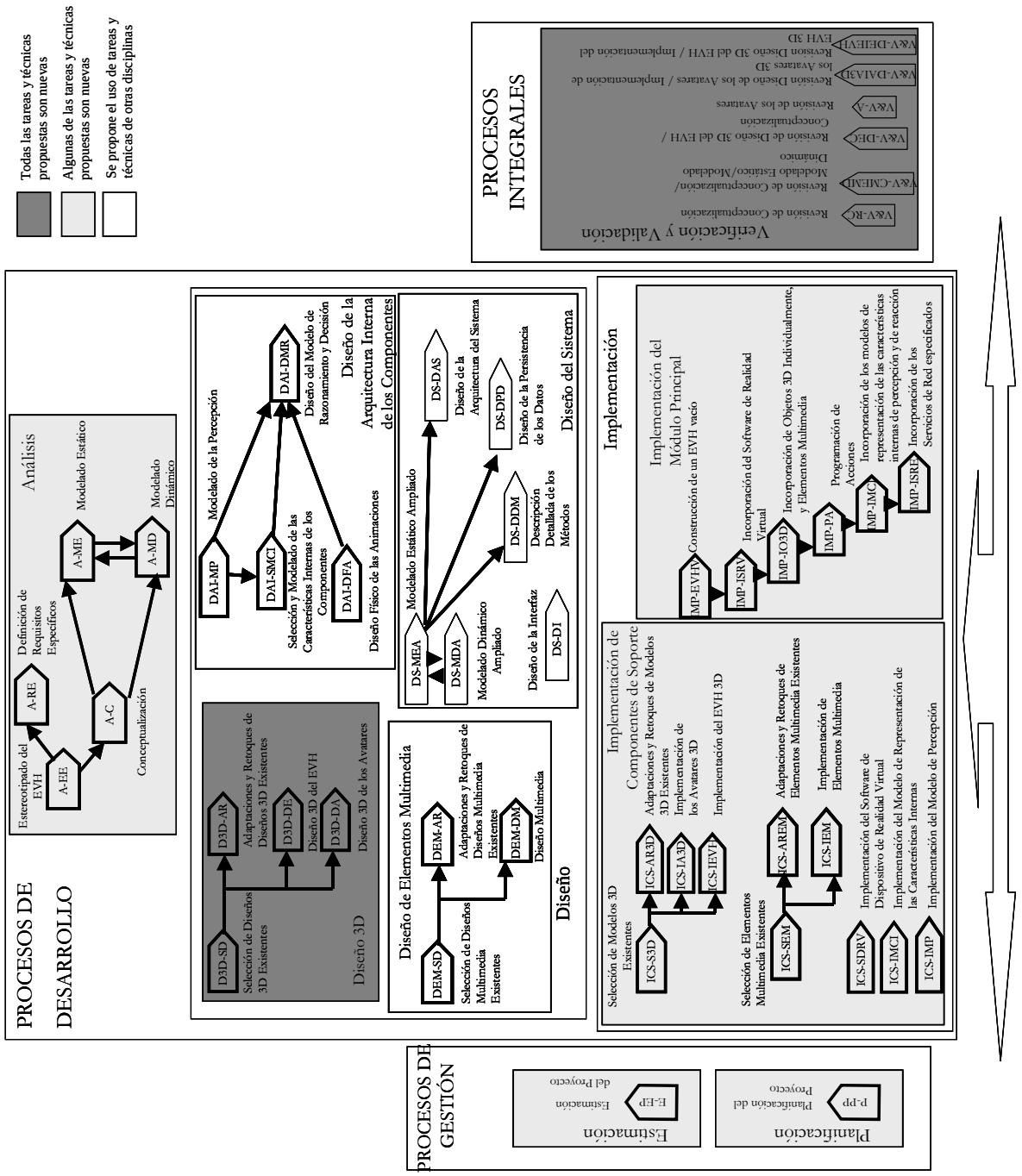


Figura 6.4. Conjunto de procesos y tareas del Marco Metodológico Propuesto

Dado que los EVs tienen muchos módulos o partes (v.g. el dispositivo de realidad virtual, los modelos 3D, elementos multimedia, etc.) que conforman todos ellos el EV, pero cuya implementación se puede llevar a cabo bien a partir de requisitos que se deciden muy al comienzo del proyecto o bien según van terminándose algunas tareas de diseño, y a su vez estos módulos no tienen relación entre sí hasta que no se integran en el EV definitivo, se ha propuesto un proceso de **Implementación de los Componentes de Soporte**, que incluye la implementación de todos estos módulos o partes de forma independiente.

A medida que dichos módulos vayan estando terminados, se irán integrando en el sistema, con el fin de ir mostrándole al cliente, lo antes posible, el EV definitivo. Para ello se ha creado un proceso de **Implementación del Módulo Principal**, que permite, de forma incremental, ir añadiendo pequeños módulos al EV. Por cada

porción del EV que se incorpore al sistema final, se le entregará una versión al usuario, de modo que si no es lo que esperaba se puedan hacer modificaciones a la parte que se acaba de incorporar.

A medida que se van realizando las tareas propuestas en SENDA es necesario ir verificando y validando los modelos que se van generando. Para ello se proponen un conjunto de tareas especiales de revisión incluidas en el proceso de Verificación y Validación.

6.3.2 El proceso de análisis en el desarrollo de EVs

Muchos investigadores sugieren que la fase de análisis debe contemplar una especificación de requisitos, o conceptualización, en la que se recojan exclusivamente características que definan qué hace el sistema, y nunca cómo debe comportarse el sistema [13]. Al igual que en [32], este trabajo comparte la perspectiva de que se trata de una idea muy atractiva, pero demasiado simplista en la práctica.

En los EVs, es especialmente relevante subrayar que se recogerán un conjunto de requisitos de muy diferente índole. Gracias a dichos requisitos se establecerán una gran variedad de características específicas del producto a desarrollar, concretamente estos requisitos permitirán determinar los valores de las características identificadas en la introducción de este capítulo y además el tipo de software que se utilizará como plataforma de desarrollo. De las decisiones que se tomen sobre dichos requisitos dependerán en gran medida otras fases del desarrollo (v.g. la estimación de coste y duración del proyecto dependerá de decisiones que se tomen en la fase de análisis).

Dentro del proceso de Análisis, se tendrán en consideración diferentes tareas que irán definiendo el sistema desde distintas aproximaciones. Como tarea inicial dentro del Proceso de Análisis se sugiere hacer un **Estereotipado del EV**, que servirá para perfilar adecuadamente las características del sistema. Para ello, se utilizarán unos cuestionarios de tipificación, cuyo contenido proporcionará al equipo desarrollador una visión más concreta sobre cuáles son las características específicas del proyecto a desarrollar, así como de las tareas que se deben realizar, ya que no todas las tareas del Marco Metodológico propuesto son necesarias para todos los proyectos.

El proceso de análisis no puede llevarse a cabo en su totalidad utilizando las técnicas tradicionales, ya que los casos de uso, utilizados en el análisis orientado a objetos, se centran sólo en las funcionalidades que el usuario puede demandar del sistema. Debido a las características de este tipo de sistemas interactivos, puede darse el caso de que algunas o muchas de las actividades que se desarrollan dentro de éste no sean demandadas por el propio usuario sino que sean automáticas. Para esto, es necesario definir los requisitos de algún modo fuera del alcance de las técnicas habituales basadas en análisis estructurado u orientado a objetos.

En el proceso de Análisis, concretamente en la tarea de conceptualización, se propone una forma adicional de definir los requisitos a través los llamados **Conceptos de Uso,** de manera que se contemple la posibilidad definir requisitos que no necesariamente impliquen la descripción de servicios que vayan a ser demandados por el usuario desde la interfaz.

Las tareas de Modelado Estático y Dinámico se han tomado de las metodologías orientadas a objetos, adaptándolas y relacionándolas adecuadamente con el resto de tareas de SENDA.

En la Tabla 6.1, aparecen las tareas de Análisis y en la Figura 6.5, la relación entre dichas tareas y los productos de tareas pertenecientes a otros procesos de SENDA.

Tabla 6.1. Tareas del proceso Análisis

	Tareas	**Acrónimo**
Proceso de Análisis	Estereotipado del EV	A-EE
	Definición de Requisitos Específicos	A-RE
	Conceptualización	A-C
	Modelado Estático	A-ME
	Modelado Dinámico	A-MD

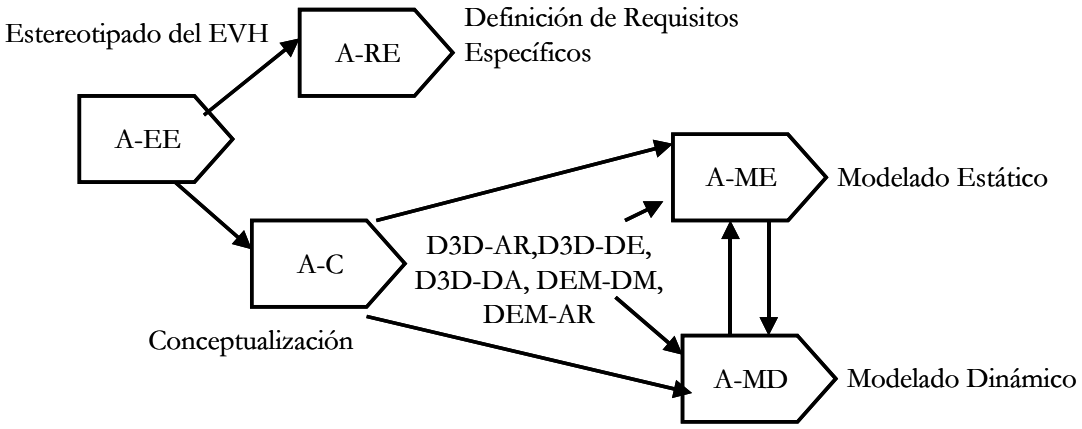


Figura 6.5. Relación entre tareas del proceso Análisis

A continuación se van a describir cada una de las tareas propuestas en el proceso de Análisis siguiendo una estructura común identificando productos de entrada de salida, así como técnicas y participantes para cada una de las tareas identificadas.

Estereotipado del EV

Se tratará de estereotipar el tipo de EV a construir a través de entrevistas con el cliente. Cada estereotipo de EV vendrá descrito por una serie de características y, como consecuencia de ello, habrá un conjunto de tareas asociadas, necesarias para llevar a cabo el desarrollo del EV de la forma más eficiente.

En la Tabla 6.2 aparecen los productos, técnicas y participantes correspondientes a esta tarea.

Tabla 6.2. Tabla de productos/técnicas/participantes de la tarea Estereotipado del EV

Productos	Entrada	Acuerdo con el cliente para iniciar el proyecto
	Salida	Estereotipo de EVa construir
		Mapa de Tareas
Técnicas		Entrevistas con el/los clientes
		Cuestionarios de tipificación
Participantes		Analista de Sistemas
		Clientes

La principal ventaja de esta tarea es la posibilidad que se le da al responsable del proyecto de construir un mapa de tareas propio para cada proyecto. Dicho mapa se puede construir a través de un conjunto de cuestiones que deben responderse sobre las características del EV que se quiere desarrollar. A partir de las respuestas del cuestionario se puede construir un mapa de tareas, a partir del cual se puede saber cuales de las tareas de SENDA han de llevarse a cabo y cuales no.

Esta tarea se propone con el fin de poder identificar, desde el principio del proyecto, el tipo de EV que se desarrollará. Además, permite al cliente asentar ideas sobre el producto que desea que le construyan, y al equipo de desarrollo tener más claras las tareas que deben llevar a cabo, y así se puede planificar mucho mejor el trabajo.

El cuestionario de tipificación que servirá para construir el mapa de tareas del EV aparece en la Tabla 6.3.

Una vez respondido el cuestionario, será responsabilidad de la persona encargada de gestionar el proyecto, construir el mapa de tareas que se llevarán a cabo. Para ello puede utilizar el mapa global que contiene todas las tareas de SENDA que parecen en la Figura 6.4.

Tabla 6.3. Cuestionario de tipificación del EV

¿El EV sólo servirá para realizar visitas guiadas, sin que exista ningún tipo de interacción?	**Sí** ☐ **No** ☐ Si la Respuesta es Sí, elimine del proceso de desarrollo el proceso de Diseño de las Características Internas de los Componentes del EV, y las tareas ICS-IMCI e ICS-IMP del proceso de Implementación de Componentes de Soporte y la tarea IMP-IMCI, del proceso de Implementación del Módulo Principal.

¿El EV funcionará en Red?	**Sí** ☐ **No** ☐ Si la respuesta es No, elimine la tarea IMP-ISRE del proceso de Implementación del Módulo Principal.
¿El EV utilizará dispositivos de realidad virtual?	**Sí** ☐ **No** ☐ Si la respuesta es No, elimine la tarea ICS-SDRV del proceso de Implementación de Componentes de Soporte y la tarea IMP-ISRV del proceso de Implementación del Módulo Principal.
¿El EV servirá para el aprendizaje?	**Sí** ☐ **No** ☐ Si la respuesta es Sí, deberá plantearse añadir a la arquitectura general de EV, un módulo tutor.
¿EL EV servirá para llevar a cabo relaciones sociales?	**Sí** ☐ **No** ☐ Si la respuesta es No, puede eliminar del proceso del proceso de Diseño de las Características Internas de los Componentes del EV, la tarea DAI-SMCI. Si la respuesta es Sí, deberá plantearse si necesita un modelo de personalidad o un modelo social para incluirlo en el EV.
¿El EV tendrá elementos 3D?	**Sí** ☐ **No** ☐ Si la respuesta es No, puede eliminar el proceso de Diseño 3D, y las tareas ICS-S3D, ICS-AR3D, ICS-IA3D, ICS-IEV del proceso de Implementación de Componentes de Soporte. Y de la tarea IMP-IO3D, recuerde que no debe realizar la parte correspondiente a la carga de elementos 3D en el EV.
¿El EV tendrá elementos multimedia?	**Sí** ☐ **No** ☐ Si la respuesta es No, puede eliminar el proceso de Diseño de Elementos Multimedia, y las tareas ICS-SEM, ICS-AREM, ICS-IEM del proceso de Implementación de Componentes de Soporte. Y de la tarea IMP-IO3D, recuerde que no debe realizar la parte correspondiente a inserción de elementos multimedia.
¿El EV tendrá avatares guiados por agentes?	**Sí** ☐ **No** ☐ Si la respuesta es Sí, los avatares deben ser modelados de tal modo que puedan ser manejados por agentes, es decir deben poder ser controlados desde la interfaz y desde dentro del sistema de forma automática, luego debe utilizar el formalismo de los Conceptos de Uso para definir algunos de los requisitos en la tarea de Conceptualización del proceso de Análisis.
¿El EV controlará total o parcialmente el modelo de personalidad para el avatar?	**Sí** ☐ **No** ☐ Si la respuesta es No, puede eliminar la tarea DAI-SMCI del proceso de Diseño de la arquitectura Interna de los Componentes y ICS-IMCI del proceso de Implementación de Componentes de Soporte.
¿El EV controlará total o parcialmente el modelo de razonamiento para el avatar?	**Sí** ☐ **No** ☐ Si la respuesta es No, puede eliminar la tarea DAI-DMR del proceso de Diseño de la arquitectura Interna de los Componentes.
¿El EV controlará total o parcialmente el modelo de percepción para el avatar?	**Sí** ☐ **No** ☐ Si la respuesta es No, puede eliminar la tarea DAI-MP del proceso de Diseño de la arquitectura Interna de los Componentes y ICS-IMP del proceso de Implementación de Componentes de Soporte.

Tarea de conceptualización
En la Tabla 6.4 aparecen los productos, técnicas y participantes correspondientes a esta tarea.

Tabla 6.4. Tabla de productos/técnicas/participantes de la tarea Conceptualización

Productos	Entrada	Estereotipo de EV
	Salida	Definición del Problema
		Definiciones, Acrónimos y Abreviaturas
		Lista inicial de requisitos funcionales del sistema
		Documento de Conceptualización, con casos de uso y conceptos de uso clasificados.
Técnicas		Conceptos de Uso
		Casos de Uso
Participantes		Analista de Sistemas
		Cliente
		Usuarios

En esta tarea se tiene que definir el problema y delimitarlo. Es importante definir claramente la magnitud del producto a construir, el ámbito en que se enmarca, el tipo de EV de que se trata, etc., teniendo en cuenta los resultados de la tarea anterior. Además, se debe hacer una lista de acrónimos y abreviaturas que se van a utilizar durante el desarrollo del sistema.

Con toda la información recopilada durante las entrevistas de la tarea de Estereotipado del EV, hay que hacer una lista en lenguaje natural que defina, en un párrafo, cada uno de los requisitos identificados, asignando para cada uno de ellos una referencia que lo distinga de los demás de forma unívoca.

Tras confeccionar la lista de requisitos anterior, se deben definir más precisamente los mismos. Para ello se utilizarán dos técnicas distintas. En el caso de los requisitos que provienen de una interacción entre el usuario y el sistema, es decir, aquellos en los que el usuario demanda una funcionalidad al sistema, se podrán refinar los requisitos utilizando para ello los Casos de Uso de Jacobson [18].

Sin embargo en los EVs se tiende a depositar cierto grado de control sobre el sistema, para que la interfaz con el usuario no esté repleta de controles que le dificulten la interacción. Será el cliente el que decida el grado de control que quiere que los usuarios tengan sobre el sistema, de manera que, si se decide delegar el control sobre el sistema, las funcionalidades ya no serán demandadas directamente por el usuario. Por este motivo los casos de uso, tal cual están definidos por [18], no sirven para especificar este tipo de requisitos. Para este tipo de requisitos este trabajo propone otro formalismo: los llamados **Conceptos de Uso**.

Un **Concepto de Uso** se redacta en una o dos frases y representa una de las posibles funcionalidades del sistema, no siendo estas funcionalidades demandadas directamente por el usuario sino delegadas en algún elemento del EV.

Determinar los conceptos de uso permite fijar rápidamente las funcionalidades que debe ofrecer el sistema al usuario. Cada concepto de uso suele coincidir con uno o varios requisitos previamente identificados. Puesto que los conceptos de uso están asociados a funcionalidades del sistema que no son demandadas directamente por el usuario, puede resultar útil describir la forma en que se deben realizar o la forma en que deben lanzarse dichas funcionalidades. Para ello, se propone asociar a cada concepto de uso un **Concepto de Operación**. Cada concepto de operación debe describir:

- El propósito: para qué se va a emplear dicho concepto de uso.
- El modo de funcionamiento: cómo se va a utilizar dicho concepto de uso.
- La frecuencia: frecuencia de utilización o frecuencia de ejecución de la funcionalidad.

Los conceptos de uso se van a representar del modo que aparece en la Tabla 6.5.

Tabla 6.5. Representación de un Concepto de Uso

CONCEPTO DE USO:	**CONCEPTO DE OPERACIÓN**
Nombre del concepto de uso: breve descripción. Código del concepto de uso: Concepto(XX), donde XX, será el número de orden en que se ha definido el concepto de uso.	Propósito: Modo de funcionamiento: Frecuencia:

De los tres atributos que definen un concepto de uso, sólo será preciso cumplimentar aquellos cuyos datos sean conocidos. Esto dependerá de los requisitos iniciales del proyecto. Los detalles de funcionamiento y frecuencia

son muy importantes para un tipo de funcionalidad identificada como Concepto de Uso, ya que será el sistema el que dispare esa funcionalidad de forma automática. Puede parecer extraño que se pida esta información en el análisis del sistema, pero en este trabajo se cree que si se conocen estos detalles, independientemente de si se está en la fase de análisis o diseño, deben salir a la luz lo antes posible y no ser guardados para un momento posterior.

En la Tabla 6.6 aparece un ejemplo de concepto de uso de la aplicación que implementa el juego del Escondite Inglés y en la Tabla 6.7 aparece un concepto de uso de la aplicación PRVIR.

Tabla 6.6. Concepto de Uso del EV del Escondite Ingles

CONCEPTO DE USO: Requisito 6	**CONCEPTO DE OPERACIÓN**
Nombre del concepto de uso: Un avatar detecta si está muy lejos o muy cerca de la pared. **Código del concepto de uso:** Concepto (2)	**Propósito:** Un determinado avatar deberá detectar su situación respecto de la pared, para así poder tomar una u otra determinación a la hora de moverse hacia ella (no correrá los mismos riesgos un avatar que esté cerca de la pared que uno que esté lejos) **Modo de funcionamiento:** En la pantalla se muestra la ubicación exacta de donde se encuentra la pared respecto del avatar. **Frecuencia:** Siempre que se mire a la pantalla deberá aparecer la situación de la pared, por lo que se tiene que actualizar constantemente.

Tabla 6.7. Concepto de Uso de la Aplicación PRVIR

CONCEPTO DE USO: Requisito 52	**CONCEPTO DE OPERACIÓN**
Nombre del concepto de uso: Atravesar torno a la entrada **Código del concepto de uso:** Concepto(14)	**Propósito:** Describir cómo debe atravesar al torno el avatar visitante cuando va a entrar a realizar un trabajo **Modo de funcionamiento:** Cuando el avatar visitante está frente al torno, para poder atravesarlo debe de introducir previamente el dosímetro y la tarjeta en la lectora, así como teclear el código de trabajo en la lectora. **Frecuencia:** Cada vez que un avatar visitante va a atravesar el torno a la entrada.

Es necesario clasificar los Conceptos de Uso y los Casos de Uso con el fin de poder, posteriormente, asignar adecuadamente métodos a clases. Para ello, se proporciona una clasificación inicial que está en función de las características generales de este tipo de sistemas. Por supuesto, sólo se utilizará lo que sea preciso de esta clasificación y, del mismo modo, pueden añadirse categorías en función de las necesidades del EV de que se trate.

En la Tabla 6.8 aparecen las diferentes categorías en que puede ser enmarcado un concepto o un caso de uso. Gracias a esta tabla el desarrollador puede agrupar las funcionalidades del sistema en tipos de interacción. Además, se podrá comprobar que están cubriéndose todas las características deseables para ese EV. También permite catalogar las funcionalidades de manera que posteriormente sea más fácil hacer su diseño e implementación (v.g. todas las funcionalidades que impliquen detección de algo están agrupadas, todas las acciones que deben representarse gráficamente también tiene su grupo correspondiente, etc.).

Tabla 6.8. Categorías generales propuestas. La presencia del simbolo * indica que no hay sub categorías identificadas

CATEGORÍA	**DESCRIPCIÓN**	**SUB CATEGORÍA:descripción**
C1	**De inicio de la conexión:** siempre y cuando se trate de un entorno multiusuario o cliente/servidor.	*****
C2	**De interfaz con dispositivos de Realidad Virtual:** siempre y cuando se haya establecido como requisito el uso de dispositivos de Realidad Virtual.	*****
C3	**De animación:** se asocian con los mecanismos de animación de los objetos reactivos y *proactivos&reactivos.*	**C3.1:** Movimiento o traslación de un elemento por el EV. **C3.2:** Expresión externa de alguna de las características

		internas de los elementos del EV. **C3.3:** De expresión de alguna acción (ya sea con el fin de interactuar o no con otro elemento del EV).
C4	**De percepción*:*** estos están relacionados con la capacidad de detectar lo que ocurre alrededor de un elemento del EV en caso de que éste sea reactivo o proactivo&reactivo. Los mecanismos de percepción o detección dentro del EV se podrían asemejar a la capacidad que tienen los humanos de percibir cosas en el mundo real.	**C4.1:** De detección de acción por parte de otro elemento del EV. **C4.2:** De detección de ubicación de otros elementos del EV. C4.3: **DE DETECCIÓN DE COLISIÓN**.
C5	**De evolución del EV:** esta categoría está relacionada con las necesidades de evolución del EV.	*
C6	**De razonamiento o decisión:** relacionadas con la actividad que se desarrolle en el EV. Seguramente, tras detectar algo, un elemento del EV debe razonar y tomar una decisión relacionada con aquello que ha detectado.	*
C7	**De comunicación con otros usuarios conectados:** en función del tipo de comunicación que van a poder establecer los usuarios a través de la aplicación. Por ejemplo, voz, *chat*, etc.	*
C8	**De Visualización de la escena**: si no existen mecanismos predefinidos para visualizar el EV, habrá que especificarlos y desarrollarlos.	*

Las categorías anteriores son generales para todos los sistemas. Una vez que se han identificado las categorías deseables para un sistema concreto, se debe rellenar una tabla cuyo formato aparece en la Tabla 6.9. Dicha tabla tiene cuatro columnas cuyo contenido se pasa a describir.

- Categoría: en esta columna aparecen todas las categorías seleccionadas para clasificar casos y conceptos de uso.
- Caso/concepto de uso que lo contempla: por cada categoría aparecerán los casos y/o conceptos de uso asociados.
- Nombre de la funcionalidad asociada: aparece por cada caso y concepto de uso el código o los códigos de los requisitos que cubre dicho caso o concepto de uso.
- Nombre de la clase en el modelo de clases: clase a la que se le asociará cada una de las anteriores funcionalidades.

Tabla 6.9. Tabla de clasificación de conceptos y casos de uso

CATEGORÍA	CASO/CONCEPTO DE USO QUE LO CONTEMPLA	NOMBRE DE LA FUNCIONALIDAD ASOCIADA	NOMBRE DE LA CLASE EN EL MODELO DE CLASES
….	….	….	….

La última columna de la tabla no se rellenará, como es lógico, en la tarea de conceptualización, sino como paso previo a la construcción del modelo de clases en la tarea de Modelado Estático.

Definición de requisitos específicos

En la
Tabla **6.10**, aparecen los productos, técnicas y participantes correspondientes a esta tarea.

Tabla 6.10. Tabla de productos/técnicas/participantes de la tarea Definición de Requisitos Específicos

Productos	Entrada	Definición del Problema
	Salida	Documento de Requisitos Específicos
Técnicas		Estudio de alternativas

	Entrevistas
Participantes	Analista de Sistemas
	Clientes
	Usuarios

Las aplicaciones basadas en EVs, requieren que se establezcan ciertos requisitos específicos en la fase de análisis. Gracias a la descripción de los requisitos específicos, el equipo de desarrollo puede ir preparando el entorno de desarrollo, e incluso cada uno de los implicados en el desarrollo puede ir entrenándose con las herramienta que tendrá que manejar llegado el momento. Es decir, los diseñadores gráficos pueden empezar a familiarizarse con la herramienta de diseño gráfico que se seleccione, los programadores con la herramienta de desarrollo y librerías, etc.

Dado que el desarrollo de EVs implica la toma de decisiones importantes y específicas al principio del proyecto, se propone un formato de documento que recoja, en fases tempranas del desarrollo, un conjunto de requisitos específicos. En la Tabla 6.11 se describen los apartados de dicho documento.

Puede parecer que es muy precipitado tomar estas decisiones al principio del proyecto, pero debe ser así con el fin de optimizar el resto de procesos, tanto de gestión como de desarrollo, ya que puede haber decisiones de hardware o software que requieran una fuerte inversión que es necesario contemplar desde el principio.

Se recomienda tomar una muestra suficientemente representativa de usuarios, y distribuirles una serie de cuestionarios sobre diferentes alternativas de interfaz. El motivo radica en el hecho de que en los EVs tiene una especial relevancia la interfaz, por lo que es mejor saber las preferencias de los usuarios antes de decidirse por una interfaz u otra. De este modo, es posible saber, por ejemplo, la predisposición que van a tener hacia el uso de un cierto dispositivo de realidad virtual. Lo ideal sería que los usuarios pudieran probar distintos mecanismos para interactuar con el sistema, como guantes de datos, casco, gafas, etc., pero esto no suele ser muy viable dado el precio de estos dispositivos.

Tabla 6.11. Descripción del documento de requisitos específicos

APARTADO	SECCIÓN	EXPLICACIÓN
1. Características de los usuarios		Es preciso indicar en este punto si los usuarios tienen algún tipo de discapacidad o preferencia que obligue a un tipo de interfaz u otro. Además, debe indicarse el rango de edad en que se encuentran los usuarios a los que va dirigido el sistema.
2. Requisitos de interfaz	Interfaz con el usuario	Los EV pueden tener multitud de interfaces diferentes, y hacer uso o no de distintos dispositivos de realidad virtual. Buena parte del sistema se verá influido por la selección de dichos dispositivos, y a su vez, los dispositivos de realidad virtual deben venir determinados por las características de los usuarios a los que va dirigido el EV, el objetivo de éste, así como el tipo de interacción requerida. Por ejemplo, si el EV está pensado para entrenar en la realización de tareas muy precisas en las que se requiere la utilización de las manos, casi seguro que la interfaz exija el uso de guantes de datos. Se debe indicar en este punto cómo será la interfaz de usuario a utilizar. Si se van a utilizar dispositivos de realidad virtual y si éstos requieren definir las características del entorno real, éstas también se describirán en este apartado.
	Interfaz con otros sistemas	Dado que se trata de sistemas complejos, es muy probable que deban existir interfaces con otros sistemas, incluyéndose aquí las interfaces con el software de los dispositivos de realidad virtual (en caso de haberlos), que generalmente es necesario desarrollar en paralelo con el propio EV. Además, puede que existan otros sistemas externos como tutores inteligentes, bases de datos, redes, etc.
	Selección del dispositivo de realidad virtual	El/los dispositivos de realidad virtual a utilizar deben quedar seleccionados en este punto. Habrá que estudiar detenidamente el tipo de usuario y el tipo de interfaz, de modo que los dispositivos seleccionados cumplan con los anteriores requisitos, en caso de haberlos.

3. Requisitos no funcionales	Requisitos hardware	Generalmente, las características de los EV hacen que los requisitos de hardware sean más estrictos que en otros sistemas. El hardware seleccionado debe ser tal que cumpla con los siguientes requisitos: Capacidad suficiente para visualizar escenas tridimensionales con suficiente calidad. Tanto la calidad de los objetos 3D como la velocidad de visualización se deben incluir como requisitos. Capacidad de funcionar en red. Capacidad de funcionar en tiempo real.
	Requisitos software	Se indicarán requisitos tanto de sistema operativo como de software, incluyendo dentro de este software el de los dispositivos de realidad virtual, el de diseño 3D, el de red y el de desarrollo propiamente dicho.
	Selección de hardware y software para el desarrollo	En función de los dos apartados anteriores se seleccionará el hardware y el software de desarrollo.
	Atributos de calidad	Si existen atributos especiales de tiempo de respuesta, rendimiento, portabilidad, fiabilidad, etc., se incluirán en este apartado.
4. Otros requisitos		En función del resultado de la tarea de estereotipado del EV, realizada en el proceso de análisis, se pueden ir definiendo con el cliente algunos aspectos relacionados con las características internas de los componentes del EV. Por ejemplo, se pueden ir decidiendo el modelo de percepción que se utilizará, las características específicas que afectarán a la arquitectura interna de dichos componentes, como pueden ser rasgos de personalidad, el estado de ánimo, el carácter, las intenciones, etc. Dichos modelos, pueden ser desarrollados explícitamente para el proyecto o ser reutilizados de otros proyectos.

Tarea de modelado estático

Esta tarea trata de obtener una imagen estática de los componentes del sistema. Estos componentes se modelarán mediante un modelo conceptual, utilizando para ello cualquier metodología basada en el paradigma de orientación a objetos [4], [19], [21], [28]. En este punto conviene recordar que esta tarea y la de Modelado Dinámico, se complementan; es decir, pueden producirse resultados en una, que afecten a la otra y viceversa, por lo que deben llevarse a cabo en paralelo.

En la Tabla 6.12 aparecen los productos, técnicas y participantes correspondientes a esta tarea.

Tabla 6.12. Tabla de productos/técnicas/participantes de la tarea Modelado Estático

Productos	Entrada	Documento de Conceptualización
		Todas las salidas del proceso de Diseño 3D
		Todas las salidas del proceso de Diseño de Elementos Multimedia
	Salida	Modelo de clases de análisis
		Tabla de clasificación de casos y conceptos de uso, de la tarea de conceptualización, ampliada.
Técnicas		Diagramas de estructura estática
Participantes		Analista de sistemas

De cada uno de los conceptos de uso, y de los casos de uso expandidos, se extraerán las clases potenciales, así como sus atributos, y las relaciones entre las clases identificadas. Para ello basta seguir cualquiera de los métodos de construcción de modelos de clases indicados en la bibliografía dedicada a la orientación a objetos [4], [28], [21], [19].

No sólo la tarea de Conceptualización contribuye a la construcción del modelo de clases. A partir de los productos de salida de los procesos Diseño 3D y Diseño de Elementos Multimedia, se podrá extraer información que aportará nuevas clases al modelo de clases. Así, los modelos 3D de los elementos que aparecerán en el EV serán clases del modelo de clases, mientras que los elementos de tipo sonido, vídeo, etc., pueden ser atributos de dichas clases.

La diferencia esencial entre las clases obtenidas a través de las funcionalidades descritas en el documento de conceptualización, y las del diseño 3D y de elementos multimedia es la que aparece en la Tabla 6.13.

Tabla 6.13. Diferencia entre clases obtenidas de diversas fuentes

Clase extraída del documento de conceptualización	Clase extraída del proceso de Diseño 3D	Clase extraída del Diseño de Elementos Multimedia
Suelen corresponderse con elementos no visibles.	Suelen corresponderse con elementos visibles (tienen representación 3D).	Suelen corresponderse con clases visibles, si se trata de imagen o vídeo, y no visibles pero sonoras, si se trata de audio.

Se propone la siguiente manera de proceder para extraer las clases y sus métodos asociados. Se extraerán los sustantivos y los verbos, de la descripción de los Casos y de los Conceptos de Uso, después se asociará a cada clase potencial, es decir, a cada sustantivo seleccionado, la funcionalidad identificada por el/los Concepto/s de Uso o Caso/s de Uso en los que ha aparecido identificada.

En este momento del desarrollo se puede volver a la tabla que se construyó en la tarea de Conceptualización, que aparece en la Tabla 6.9, y rellenar la columna que indica el nombre de la clase a la cual atribuimos el/los Concepto/s o el/los Caso/s de Uso.

Como ayuda a la identificación de clases se puede decir que serán clases potenciales los siguientes elementos del EV:

- El propio EV.
- Avatares
- Cerebro del avatar: donde se podrían ubicar los mecanismos de razonamiento y decisión.
- El cuerpo del avatar
- Por cada característica interna de los componentes del EV que se componga de más de un elemento, se construirá una clase. Es decir, si una característica interna es el humor, y éste se compone de grado de alegría y grado de enfado, por depender de más de un subelemento construiremos la clase humor.
- Todos aquellos objetos gráficos que tengan comportamiento.
- Si existen dispositivos de realidad virtual se debe incluir una clase gestora de éstos.
- Para poder almacenar la historia pasada de un avatar en el EV, es necesario definir la clase Memoria.
- Los mecanismos de percepción conviene agruparlos en clases que pueden recibir el nombre de los sentidos, en función de lo que sean capaces de percibir.
- Punto de vista para poder cambiar la posición desde la que se ve el EV.

También se propone una lista de relaciones típicas que se suelen dar entre elementos de un EV:

- El avatar tiene cuerpo, para poder expresar gráficamente las acciones.
- El avatar tiene cerebro, siempre que pueda tomar decisiones.
- El avatar tiene memoria, en la cual podrá guardar su historia pasada en el EV.
- El usuario puede tener al mismo tiempo varios dispositivo de realidad virtual seleccionados.
- El EV tiene elementos, que pueden ser de los tipos de la clasificación de componentes propuesta en este trabajo.
- El usuario tiene en cada momento un único punto de vista sobre el EV.
- El usuario puede usar varios dispositivos de conexión a la vez.

El modelo básico de clases que se propone es el que aparece en la Figura 6.6.

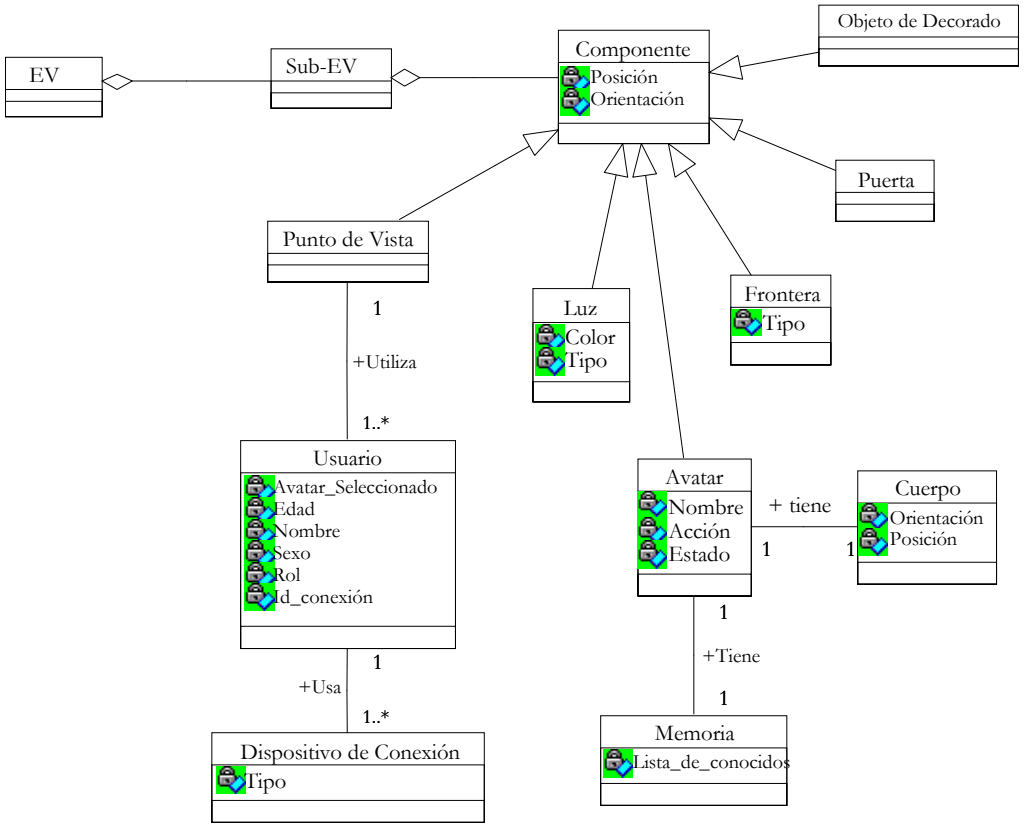


Figura 6.6. Modelo básico de clases

La Figura 6.7 representa el modelo de clases de la aplicación PRVIR.

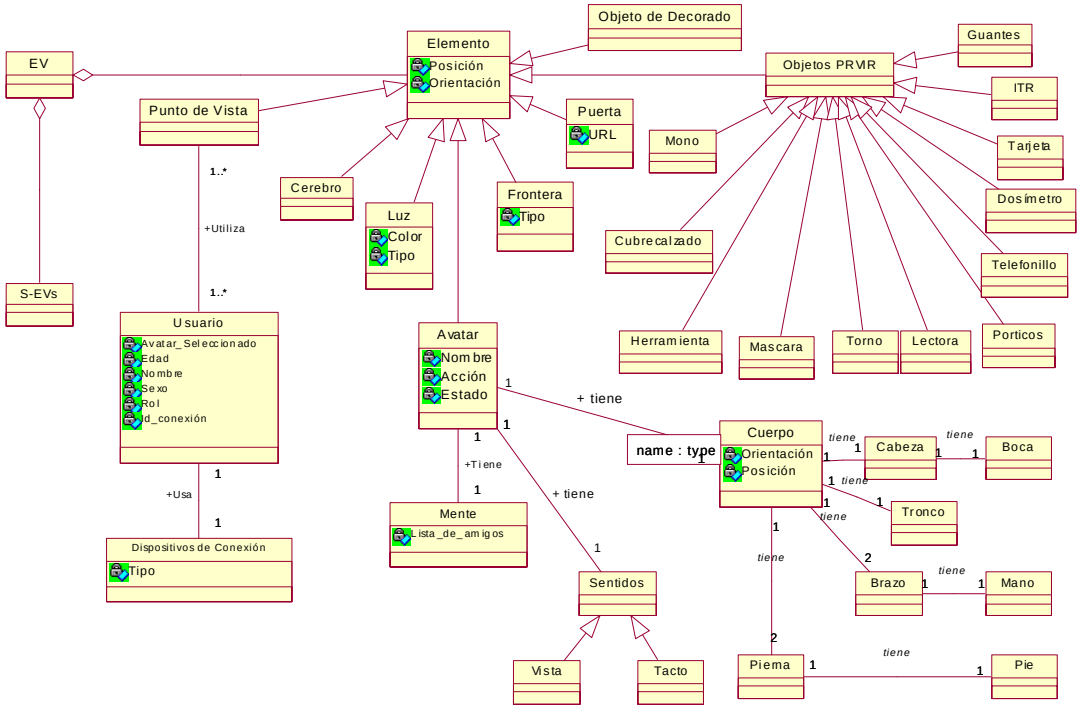


Figura 6.7. Modelo de clases de la aplicación PRVIR

Tarea de modelado dinámico

En este punto del desarrollo, la parte dinámica del sistema que puede analizarse varía sobre todo en función de la cantidad de información que se haya podido conseguir para construir los Conceptos de Uso. Cuanta más información se tenga, con mayor detalle se podrá describir la dinámica del sistema.

En la Tabla 6.14 aparecen los productos, técnicas y participantes correspondientes a esta tarea.

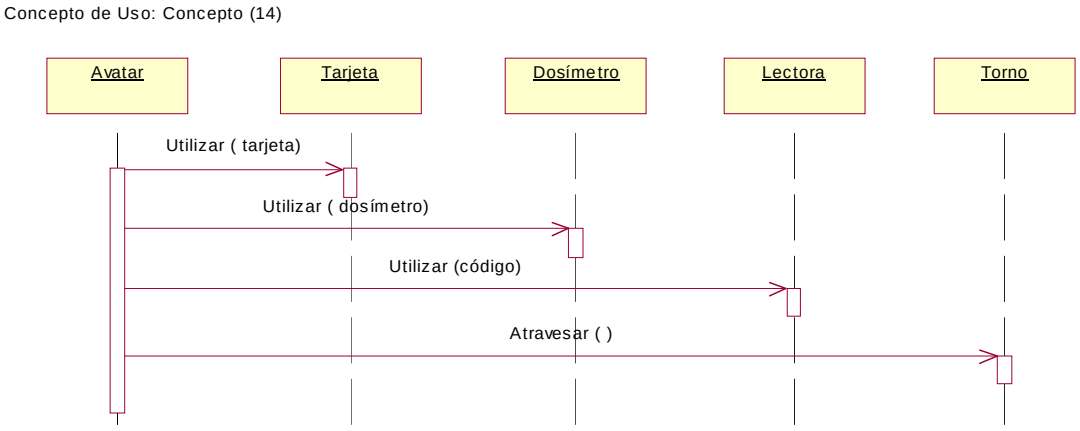
Tabla 6.14. Tabla de productos/técnicas/participantes de la tarea Modelado Dinámico

Productos	Entrada	Documento de Conceptualización
		Modelo de Clases de Análisis
	Salida	Modelo Dinámico
Técnicas		Diagramas de Secuencia del Sistema
		Escenarios
Participantes		Analista de Sistemas

Para los Casos de Uso obtenidos en el análisis, el tratamiento puede ser, por ejemplo, el propuesto en [21]: a partir de ellos se construyen los Diagramas de Secuencia del Sistema y los Contratos de Operación para describir el comportamiento del sistema.

En cambio, en el caso de los Conceptos de Uso no se puede utilizar directamente el método de Larman, ya que no los contempla. Los Diagramas de Secuencia del Sistema deben ser modificados de manera que sea una clase del sistema la que inicia los comportamientos, en lugar de un actor externo.

En la Figura 6.8 se incluye la descripción de la dinámica del Concepto de Uso de la aplicación PRVIR que aparece en la Tabla 6.7 en el que el Avatar se disponía a atravesar el torno a la entrada de la central nuclear.



DOCUMENTACION	
Avatar	Se corresponde con la representación del usuario dentro del entorno virtual.
Tarjeta	Es una tarjeta que el avatar visitante posee para su identificación.
Dosímetro	Se corresponde con el dosímetro empleado por un avatar visitante en sus tareas dentro de la zona radiológica
Lectora	Se corresponde con la lectora situada junto al torno para verificar el acceso del avatar visitante a la zona radiológica.
Torno	Se corresponde con el torno que da acceso a la zona radiológica.
Acciones	Utilizar(tarjeta): El avatar utiliza la tarjeta para que sea leída por la lectora.
	Utilizar(dosímetro): El avatar utiliza el dosímetro para que sea leído por la lectora.
	Utilizar(código): El avatar teclea su código de trabajo a realizar.
	Atravesar( ): El torno gira para que pueda ser atravesado.

Figura 6.8. Descripción dinámica del Concepto de Uso de la Tabla 6.7

Para las clases que representan las Características Internas de los Componentes del EV, es conveniente construir un Diagrama de Transición de Estados, ya que este tipo de diagramas es especialmente útil para clases cuyo comportamiento no sea trivial, como ocurre con las clases que representan características internas de los componentes del sistema.

6.4 Posibilidades de las aplicaciones de los Entornos Virtuales

Los entornos virtuales se han venido aplicando en numerosos dominios, de los cuales mencionaremos algunos ejemplos destacados. Las principales limitaciones para llevar estas aplicaciones a la web vienen dadas, hoy en día, por las restricciones de ancho de banda en la comunicación por la red, y por las limitaciones para el uso de dispositivos de realidad virtual inmersivos y de realidad aumentada. El hecho de funcionar sobre Internet puede plantear sobre todo problemas de velocidad, de seguridad y de fiabilidad. Estos problemas de velocidad, a su vez, pueden obligar a reducir el realismo y la interactividad del sistema. No obstante, son numerosos los ejemplos de entornos virtuales que se han desarrollado y utilizado con éxito para la web, en diferentes dominios de aplicación. Todo depende de los requisitos de cada sistema. En general cabe afirmar que cuanto mayores sean los requisitos de realismo de los modelos tridimensionales del entorno, y cuanto más compleja sean las posibilidades de interacción que el entorno debe ofrecer a sus usuarios, más difícil será conseguir un buen resultado a través de web. Así, las aplicaciones en medicina y en ingeniería suelen ser las más críticas.

6.4.1 Aprendizaje

Esta es quizá la aplicación más frecuente en la actualidad de los entornos virtuales y la realidad virtual. [23] [26].

Como entornos tridimensionales, se han aplicado con más éxito a situaciones de entrenamiento que a situaciones de formación. Estos entornos permiten a los estudiantes navegar e interactuar con una representación virtual de algún entorno real en el cual deben aprender a realizar alguna tarea. Son especialmente útiles cuando el entorno real no está disponible para el entrenamiento, o bien es muy costoso o arriesgado (v.g. una central nuclear [36]). Si el entorno virtual es multiusuario, permitirá además el entrenamiento de equipos de trabajo [1].

Un caso especial de entorno virtual para la formación podemos encontrarlo en algunas de las actuales plataformas de *e-learning*, que ofrecen a los estudiantes la posibilidad de reunirse virtualmente con otros estudiantes para discutir sobre las materias en estudio, o bien que ofrecen entornos de clases virtuales en las que el profesor imparte una clase de forma remota. Sin embargo, los estándares actuales de *e-learning* no están preparados para la inclusión, dentro del material formativo, de entornos virtuales tridimensionales, si no es como un objeto de aprendizaje completo e indivisible.

6.4.2 Trabajo colaborativo

Una de las aplicaciones más extendidas de los entornos virtuales, especialmente de los entornos multiusuario, es el trabajo colaborativo (*CSCW – Computer Supported Collaborative Work*). [2] [27]

Estos entornos ofrecen a los usuarios herramientas para compartir información, comunicarse y colaborar en la realización de determinadas tareas. Mediante la representación virtual de los demás usuarios, se facilita la toma de conciencia de la actividad que realizan los distintos miembros del equipo.

6.4.3 Medicina

La medicina es uno de los dominios en los que las aplicaciones de los entornos virtuales resultan de utilidad indiscutible, tanto para la práctica diaria de la medicina como para la formación de los futuros médicos [9] [35].

Así podemos encontrar aplicaciones de realidad aumentada en las que datos del paciente obtenidos mediante técnicas como la Tomografía Axial Computerizada o la Imagen por Resonancia Magnética son combinados con la vista del paciente real, dotando al médico de una especie de “visión de rayos X”; aplicaciones en las que los estudiantes de medicina intervienen a un paciente virtual con cirugía laparoscópica; etc.

6.4.4 Simulación, evaluación y predicción

Algunos EVs han sido desarrollados para simular determinados sistemas reales con el propósito de estudiar los mecanismos que regulan el funcionamiento de dichos sistemas, o bien para poder someterlos a determinadas

situaciones simuladas, estudiar la respuesta del sistema y evaluar su comportamiento, o bien para predecir cuál será el comportamiento del sistema real ante determinadas circunstancias.

Ejemplo destacado y especialmente interesante de estos sistemas son los que simulan multitudes humanas. Se han utilizado, por ejemplo, para simular la evacuación de una multitud de un gran edificio y evaluar diferentes tipos de sistemas de seguridad, caminos de evacuación, etc. [33].

6.4.5 Entretenimiento

Las posibilidades de los EVs en el mundo del entretenimiento son casi infinitas. Podemos encontrar entornos virtuales multiusuario en los que los visitantes pueden participar en actividades lúdicas tan simples como pasear y conversar con otros visitantes [34], o tan complejas como asistir a representaciones teatrales o de danza virtual [24].

Podemos encontrar aplicaciones de realidad aumentada donde un personaje virtual juega con nosotros al ajedrez o a un juego de cartas, o podemos hacer visitas turísticas virtuales a lugares lejanos, a reconstrucciones virtuales de lugares ya desaparecidos, o a sitios fantásticos donde podemos vivir todo tipo de aventuras.

6.4.6 Aplicaciones de los agentes virtuales

Un apartado independiente merecen las posibles aplicaciones de los agentes virtuales, que incluyen su participación en EVs culturales actuando como guías turísticos, en EVs de comercio electrónico actuando como vendedores, en EVs de formación y entrenamiento actuando como profesores, en EVs de entretenimiento como compañeros de juego, etc.

Esta es seguramente una de las áreas de los entornos virtuales más abiertas a la investigación, ya que en la actualidad aún estamos lejos de poder disponer de agentes virtuales realmente flexibles, inteligentes, y con potentes capacidades de interacción con los humanos [10]. En este sentido cabe destacar los trabajos sobre agentes virtuales conversacionales capaces de entablar diálogos con seres humanos [6], y las investigaciones sobre agentes pedagógicos [26].

6.5 Conclusiones

A lo largo de este capítulo se han descrito conceptos básicos relacionados con los EVs, de modo que tanto los noveles como los veteranos en esta área puedan empaparse de una terminología común utilizada a lo largo del resto del capítulo.

Dado el carácter multidisciplinar de este tipo de aplicaciones, de la carga de conceptos que puede albergar; sociales, psicológicos, etc., así como de las posibilidades interactivas que presentan, han sido durante los primeros años de su aparición objeto de una evolución tecnológica muy rápida. Esto es normal siempre que surge un tipo de aplicaciones novedosas y que brinden tantas posibilidades. Es por esto por lo que se construyeron multitud de plataformas de desarrollo, hablando en términos de implementación, algunas de las cuales hemos presentado al lector.

Una vez resueltas las cuestiones puramente técnicas, la disciplina de la Ingeniería del Software se presta como el vehículo idóneo para aunar experiencias en diferentes áreas: Diseño Gráfico, Interacción Persona Ordenador, Psicología, Sociología, etc., y definir de forma ordenada, rigurosa y repetible el modo y las técnicas que pueden asegurar el éxito en la construcción de un EV. El marco metodológico SENDA abarca el desarrollo completo de EVs, independientemente de la tecnología adoptada en cada caso. En este capítulo hemos querido resaltar el proceso de Análisis descrito en SENDA, proporcionando además ejemplos de aplicación del proceso de Análisis en proyectos reales.

Por último, tras exponer nuestra experiencia en este tipo de desarrollos, hacemos un recorrido por las principales áreas en las que los EVs están teniendo aplicación. Si bien esta breve revisión permite augurar un gran éxito, es cierto que aún nos enfrentamos a problemas a la hora de desarrollar estos sistemas, debidos algunos de éstos a cuestiones tecnológicas que, esperemos, poco a poco se irán resolviendo, pero también debidos a carencias metodológicas que en cierta medida hemos intentado paliar con este trabajo. También resaltar el hecho de que diseñar EVs para Web o sin Web es exactamente lo mismo pero actualmente existen problemas tecnológicos y de compatibilidad que dificultan el uso de estas tecnologías en la Web. Tal vez la relación de Web con televisión digital podría ser su punto de fusión que terminaría con muchos de los problemas tecnológicos actuales de desarrollo de EVs para Web.

Referencias

1. de Antonio, A., Ramírez, J., Méndez, G. (2004). An Agent-Based Architecture for Virtual Environments for Training. En Sánchez-Segura, M. (ed.) Developing Future Interactive Systems. Idea Group Publishers.
2. Benford, S., Fahlén, L. E. and Bowers, J. M. (1994) Managing Mutual Awareness in Collaborative Virtual Environments, en Singh, G.,Feiner, S. K., Thalmann, D. (eds.) Proceedings ACM SIGCHI Symposium on Virtual Reality Software and Technology (VRST'94), pp.223-236, World Scientific: Singapore
3. Benford, S., Greenhalgh, C., Lloyd, D. (1997) Crowded collaborative virtual environments. Proceedings of the SIGCHI Conference on Human factors in computing systems, p.59-66, March 22-27, 1997, Atlanta, Georgia, United States.
4. Booch, G. (1993). Object-oriented analysis and design with applications. Addyson-Wesley.
5. Bricken, M. (1990). Virtual Worlds: no Interface to Design. Human Interface Technology Center. University of Washington. Technical Report R-90-2.
6. Cassell, J.; Pelachaud, C.; Badler, N.; Steedman, M.; Achorn, B.; Becket, T.; Douville, B.; Prevost, S.; and Stone, M. (1994). Animated conversation: Rule-based generation of facial expression, gesture and spoken intonation for multiple conversational agents. Proceedings of ACM SIGGRAPH '94. pp. 413-420
7. Frécon, E., Stenius, M. (1998). DIVE: A scalable network architecture for distributed virtual environments. Distributed Systems Engineering Journal (DSEJ) Vol. 5 , pp. 91-100, Special Issue on Distributed Virtual Environments.
8. Frécon, E. (2004) DIVE: Communication Architecture and Programming Model, IEEE Communications Magazine, Vol. 42, No. 4, April 2004.
9. Fuch, H. y otros. Augmented Reality Visualization for Laparoscopic Surgery. 1st International Conference Medical Image Computing and Computer-Asisted Intervention (MICCAI 98), Springer-Verlag, Heidelberg, Germany 1998, pp. 934-943.
10. Gratch, J., Rickel, J. et al (2002) Creating Interactive Virtual Humans: some assembly required. IEEE Intelligent systems, july/august 2002, pp.2-11.
11. Greenhalgh, C. M., and Benford, S. D. MASSIVE: A Virtual Reality System for Tele-conferencing, ACM Transactions on Computer Human Interfaces (TOCHI), 2 (3): 239-261, ISSN 1073-0516, ACM Press, September 1995.
12. Greenhalgh, C., Purbrick, J., Snowdon, D. (2000) Inside MASSIVE-3: Flexible Support for Data Consistency and World Structuring. Proceedings of the Third International Conference on Collaborative Virtual Environments (CVE 2000), pp. 119-127, ACM ISBN 1-58113-303-0. September 10-12 2000, San Francisco, CA.
13. Hsia, P, Davis, A., Kung, DC. (1993). *Status report: requirements engineering*. IEEE Software. Vol 10. Nº 6. pp. 75-79.
14. ISO/IEC 14772-1:1997. Information technology -- Computer graphics and image processing -- The Virtual Reality Modeling Language -- Part 1: Functional specification and UTF-8 encoding
15. ISO/IEC FDIS 14772-2:2004. Information technology -- Computer graphics and image processing -- The Virtual Reality Modeling Language (VRML) -- Part 2: External authoring interface (EAI)
16. ISO/IEC 19775-1:2004. Information technology - Computer graphics and image processing - Extensible 3D (X3D) - Part 1: Architecture and base components
17. ISO/IEC FCD 19774. Information technology - Computer graphics and image processing - Humanoid Animation (H-Anim)
18. Jacobson, I., et al. (1992) Object Oriented Software Engineering: A Use Case Driven Approach. Addison-Wesley.
19. Jacobson, I., Booch, G., Rumbaugh, J. (1999). The Unified Software Development Process. Addison-Wesley.
20. Kruchten, P. (1999). The Rational Unified Process. An Introduction. Addison-Wesley Object Technology Series.
21. Larman, C. (1998). Applying UML and patterns: an introduction to object-oriented analysis and design. Prentice Hall.
22. Macedonia, M., Pratt, D., & Zyda, M. (1994). NPSNET: A network software architecture for large scale virtual environments. Presence: Teleoperators and Virtual Environments, 3(4):265-287. Cambridge MA: MIT Press
23. McKay, D.P., Matuskey, P., Testani, S., et al. (1998). An Architecture for training virtual worlds environments. Proceedings of the virtual worlds and simulation conference. VSIM'98. Society for computer simulation, San Diego, USA
24. Meador, W. S., Kurt, E. M., O'Neal, K. R. (2003) Virtual performance and collaboration with improvisational dance. Proceedings of the SIGGRAPH 2003 conference on Sketches & applications: in conjunction with the 30th annual conference on Computer graphics and interactive techniques, San Diego, California
25. Milgram, P., Takemura, H., Utsumi, A., Kishino, F. (1994) Augmented Reality: A Class Of Displays On The Reality-Virtuality Continuum. SPIE Vol. 2351, Telemanipulator and Telepresence Technologies
26. Rickel, J., Johnson, W.L. (1999) Animated agents for procedural training in virtual reality: Perception, cognition and motor control. Applied Artificial Intelligence 13, pp.343–382.
27. Rodden, T. (1996) Populating the Application: A Model of Awareness for Cooperative Applications. ACM Conference on Computer Supported Cooperative Work (CSCW'96), Boston, Massachusetts, USA, ACM Press, pp. 87-96.
28. Rumbaugh, J. (1991). Object-Oriented Modeling and Design. Prentice-Hall International.
29. Sánchez-Segura, M. (2001). Aproximación Metodológica al Desarrollo de Entornos Virtuales Ph. D. Thesis. Technical University of Madrid. Spain.
30. Sánchez-Segura, M., et al. (2004). Developing Future Interactive Systems. Idea Group Publishers.
31. Sánchez-Segura, M., de Antonio, A, Amescua, A., (2004). Interaction patterns for future interactive systems components, Interacting with Computers (Vol 16/2 pp 331-350)
32. Sommerville, I., Sawyer, P. (1997). Requirements Engineering: a good practice guide. Wiley and Sons.
33. Thompson, P.A., Marchant, E.W. (1995) A Computer Model for the Evacuation of Large Building Populations. Fire Safety Journal, n. 24, pp-131-148.
34. Waters R.C., Anderson, D. B., Barrus, J. W., Brogan, D. C., Casey, M. A., McKeown, S. G., Nitta, T., Sterns, I. B., Yerazunis, W. S. (1997). Diamond Park and Spline: A Social Virtual Reality System with 3D Animation, Spoken Interaction, and Runtime Modifiability. MERL TR 96-02. Presence: Teleoperators and Virtual Environments 6(4): 461-480. Cambridge MA: MIT Press

35. Weghorst, S. Augmented Reality and Parkinson's Disease. Communications of the ACM, 40(8)
36. Youngblut, C. (1998). Educational Uses of Virtual Reality Technology. IDA Document D-2128. Alexandria, VA: Institute for Defense Analyses