EVOO: una Arquitectura Básica para la Construcción de Mundos Virtuales

Máximo Perez Villalba
Lifia, Facultad de Informática, UNLP
cc11, La Plata, B1900BAW, Argentina
(+54 221) 423 6585
villalba@sol.info.unlp.edu.ar

Alicia Díaz
Lifia, Facultad de Informática, UNLP
cc11, La Plata, B1900BAW, Argentina
(+54 221) 423 6585
alicia@sol.info.unlp.edu.ar

RESUMEN
En este artículo se describe un frameworks que permite interconectar un ambiente virtual clásico con una interfaz inmersiva 3D. Las EVOO (Entidades Virtuales Orientadas a Objetos) son una nueva tecnología para la construcción de simulaciones de mundos virtual, la cual modela un objeto virtual cualquiera que puede aparecer en un mundo virtual; y además, combina los conceptos de comportamientos físicos y espaciales de las tecnologías gráficas tridimensionales y el concepto de comportamiento conceptual al estilo de los ambientes virtuales clásicos. Las EVOOs están pensadas como un intermediario entre una aplicación real de software, dónde esta descrito el comportamiento propio del dominio de la aplicación, y una tecnología de gráfica tridimensional como interfaz, ya que estas hacen un mejor aprovechamiento de los recursos que provee.

Palabras Claves
Realidad Virtual, Mundo Virtual, interfaces 3D, Entidades Virtuales, Orientación a Objetos

1 INTRODUCCIÓN
Las recientes investigaciones desarrolladas tanto en el área de realidades virtuales (RV) como en trabajo cooperativo soportado por computadoras (CSCW: Computer Supported Cooperative Work) han evolucionado hacia una nueva área de investigación: los *Ambientes Virtuales Cooperativos* (CVE: Cooperative Virtual Environments). Un CVE es un espacio virtual donde los usuarios en forma distribuida colaboran y cooperan entre ellos, o con otras componentes de software, de manera sincrónica o asincrónica, en el contexto de un mundo virtual que emula algún aspecto físico del mundo real [7]. Un CVE también es caracterizado por:

- Un *espacio virtual*: la mayoría de los CVE se valen de una metáfora espacial como un edificio, un conjunto de habitaciones interconectadas o cualquier otra para ubicar sus componentes: usuarios u otros objetos virtuales. Esta metáfora es lo que se llama *espacio virtual* [1]. Cabe destacar la diferencia entre los siguientes dos términos “lugar” y “espacio”. El lugar es la ubicación donde la acción sucede, más que la representación física del espacio [9]. Las aplicaciones MOOs (como más adelante vamos a detallar) se basan en el concepto de lugar, en cambio las aplicaciones de RV se basan en los conceptos de espacio.
- Los *habitantes*: el espacio virtual esta poblado por objetos virtuales [2]. Algunos ejemplos de ellos son un robot, un *shopping basket*, un documento, una mesa de reunión, un pizarrón o incluso un colectivo. También los usuarios son entendidos como habitantes del espacio virtual.
- Otra característica distintiva de los CVE es que el usuario tiene una representación física en el ambiente virtual. Esta representación suele llamarse *character* o *avatar* [3] en las RVs.
- Tanto los usuarios como los demás objetos pueden moverse en el espacio virtual con el objeto de encontrarse y interactuar con otros usuarios u objetos.

Con formato

En la actualidad existen muchas aplicaciones de mundos virtuales con distintos objetivos, quizás las mas difundidas son aquellas para trabajo cooperativo, para educación a distancia, para comercio electrónico o simplemente para entretenimiento. Todas ellas de alguna manera emulan algún aspecto del mundo real.

Muchas de las aplicaciones anteriormente mencionadas usan diferentes interfaces para digitalizar el mundo:

Por un lado tenemos aquellas que hacen uso de la tecnología de realidad virtual para construir sus interfaces, como por ejemplo usando: VRML, Active Worlds o Viscape. *VRML* (Virtual Reality Markup Language) [11] extiende la web para soportar gráficos tridimensionales, donde los "mundos" construidos con VRML contienen objetos (figuras tridimensionales) que pueden actuar como links a otros documentos en la web, incluso a otros "mundos" VRML. En *Active Worlds* el usuario puede participar en encuentro con otros usuarios provenientes de cualquier lugar, construir su propio hogar (*virtual home*), participar de juegos *online*, navegar en la web en 3D, etc. (http://www.activeworlds.com). Otra alternativa es *Viscape*, un navegador para páginas 3D (http://www.superscape.com).

Por otro lado tenemos los ambientes MOOs (Multi-user Domain/Dungeon Object-Oriented Environments). Un MOO es una realidad virtual basada en texto, multi-usuario y programable [8]. Los MOOs son muy fáciles de configurar y con una amplia capacidad para el trabajo sobre redes de computadoras. Estas características los han convertidos en una elección válida para la implementación de sociedades virtuales y sistemas de conferencias, incluso para implementar juegos tan tradicionales como sus antecesores los MUDs [5]. En los ambientes MOOs varios usuarios se pueden comunicar entre ellos en tiempo real, moverse dentro del espacio virtual conformado por un conjunto de habitaciones (*rooms*) conectadas entre sí, e interactuar con otros objetos que se encuentran en las habitaciones. Cada habitación tiene asociada una descripción textual y puede contener uno o más habitantes, cada uno con su propia descripción textual. Los MOOs crecen continuamente gracias al esfuerzo de los usuarios que pueden crear nuevas habitaciones, conectarlas y crear incluso otros objetos que ubican en las habitaciones, todo esto a través de la escritura de programas y participando en la organización de la comunidad. LambdaMoo [10], uno de los MOO pioneros, tiene su propio lenguaje de programación: *LambdaMoo language*, que es relativamente pequeño, orientado a objeto y de muy fácil aprendizaje para no programadores. En la sección 2 el lector encontrara más detalle del modelo conceptual que proponen los ambientes MOOs

Ambos tipos de ambientes están diseñado para implementar ambientes virtuales en los cuales existen objetos o cosas virtuales y usuarios que los recorren, los cuales pueden interactuar con las cosas u objetos que están dentro del espacio, donde un espacio virtual puede conectarse a otros espacios virtuales y formar un mundo virtual. Pero, por un lado las tecnologías de RV son mucho más adecuadas a la hora de construir buenas interfaces inmersivas, recreando más adecuadamente una "realidad" desde el punto de vista de la percepción de las figuras, su volumen y movilidad (comportamiento físico y espacial). En cambio, los MOOs no son tan buenos a la hora de definir interfaces, pero son mucho más apropiados para definir comportamientos más complejos y permiten diseñar conceptualmente el ambiente virtual en función del comportamiento propio de cada componente de su modelo subyacente.

Inspirados en ambas tecnologías, el objetivo de este trabajo fue desarrollar una arquitectura que nos permita construir mundos virtuales, no solo desde el punto de vista de las interfaces 3D, sino también del comportamiento conceptual. Para ello se diseñaron las *Entidades Virtuales Orientadas a Objetos* (EVOO) como una componente que permite modelar en un mismo objeto tanto el

comportamiento físico y espacial, como el que se provee en interfaces tipo VRML, como el comportamiento conceptual al estilo de los MOOs.

En la sección 2 se detallan los distintos tipos de comportamientos involucrados en un ambiente virtual: comportamiento conceptual, espacial y físico. En las secciones 3 y 4 se presenta el modelo conceptual y la arquitectura básica de una EVOO, respectivamente. En la sección 5 se analiza como una EVOO modela los tres tipos de comportamiento: físico, espacial y conceptual; y en la sección 6 se muestra un caso de su uso. Finalmente, la sección 7 presenta las conclusiones y trabajos futuros.

2 CARACTERIZACIÓN DE LOS AMBIENTES VIRTUALES

En los ambientes virtuales la diversidad de objetos virtuales que se pueden encontrar es incalculable, pero básicamente estos responden a la siguiente clasificación: espacios, usuarios y otros objetos virtuales. Si bien en un principio esta clasificación pareciera muy simplificada, luego se la puede enriquecer asociándoles propiedades como la movilidad, contenedores, autonomía, etc. Se define como *entidad* a cualquier cosa que conforme el ambiente virtual. Y *simulación* a una aplicación de realidad virtual particular.

Independientemente del tipo de entidad que sea, en un mundo virtual cualquier entidad exhibe tres tipos de comportamientos:

- *Comportamiento espacial:* permite a una entidad moverse, desplazarse o ser desplazada dentro de un espacio virtual.
- *Comportamiento físico:* es aquel que le permite a una entidad conocer cual es su forma o figura tridimensional y de poder comportarse como un cuerpo sólido.
- *Comportamiento conceptual:* es todo aquel comportamiento que no es ni físico ni espacial, que se le puede agregar a una entidad para particularizarla o darle cierto grado de “inteligencia”.

Esta clasificación de los diferentes tipos de comportamiento permite sentar las bases para organizar y construir un ambiente virtual. Además, permite hacer un análisis más acabado de las tecnologías existentes para la construcción de ambientes virtuales, donde algunas hacen un especial hincapié en la modelización gráfica del ambiente, en donde las entidades poseen comportamientos físicos y espaciales; en cambio otras tecnologías diseñan el ambiente basado en un modelo subyacente que permite definirle a las entidades comportamiento conceptual y espacial.

El caso de los MOOs. Esta tecnología modeliza el ambiente desde un punto de vista conceptual y espacial. El ambiente virtual es modelado sobre un espacio virtual, el cual generalmente es descompuesto en espacios virtuales más pequeños, donde la unidad mínima de espacio es llamada *room*. Dentro de los *rooms* es en donde se encuentran todos los objetos virtuales y por consecuencia donde ocurren todas las acciones o interacciones. Los *room* pueden estar conectados a otros *rooms* por medio de un objeto virtual llamado *gate* y a través de estos un objeto virtual puede navegar por los distintos *rooms* conectados. Los objetos virtuales son los que pueblan los *rooms*. Un tipo especial de objeto virtual es el que representa a los usuarios dentro del *room*, este es llamado *character* y es manipulado directamente por el usuario. Dentro de un MOO un *character* sabe de la existencia de un objeto virtual, por que el room le describe su entorno para que éste pueda interactuar. Un *character* posee características de comportamiento espacial ya que este puede desplazarse entre *rooms*, pero carece de características de comportamiento físico por no tener una forma, figura o cuerpo que lo represente dentro del ambiente, esto es debido a que la tecnología de MOOs está construida para interactuar a través de una interfaz tipo texto, por lo que no es necesario que los objetos virtuales dentro de un *room* contemplen características de comportamiento físico, ellos sólo poseen una descripción textual de sí mismos. Los *characters* también pueden tomar cualquier objeto virtual de

un *room* y llevárselo a otro, sin tener que pensar en las dimensiones del objeto virtual y si éste pasa por la puerta que conecta ambos *rooms* ya que la puerta tampoco tiene dimensiones. Una ventaja de los MOOs es la de permitir que si hay más de un *character* dentro de un mismo *room*, estos puedan comunicarse entre sí. Otra ventaja es la facilidad de personalización del ambiente que les provee a los usuarios.

En el caso de los ambientes virtuales construidos con tecnología de gráficos 3D, vemos que estas tecnologías grafican el entorno para que el usuario lo pueda ver, de esa manera permiten que pueda interactuar con lo que ve. Las tecnologías manejan una descripción gráfica 3D de figuras las cuales pueden tener características de comportamientos físicos y espaciales. Las figuras son las que pueblan el espacio virtual sobre la que se basa la simulación. En la tecnología de gráficos 3D los espacios virtuales son únicos, por lo que no están descompuestos en espacios más pequeños. Estos a priori son infinitos y vacíos. Tampoco permiten contener otros espacios, sólo contienen figuras las cuales ocupan una parte del espacio que las contiene. Los espacios pueden conectarse a otros por medio de *web links*. Los usuarios también son representados por una figura, y son llamados *avatars*. Estos tipos de tecnologías se centran solamente en la modelización del *display* de las figuras, de que éstas tengan una actualización en tiempo real y en posibilitar que ellas puedan recibir eventos generados a través del *mouse* o del teclado (comandos no textuales). Dejando casi totalmente de lado la modelización conceptual de la figura. Una de las tecnologías de construcción de ambientes gráficos 3D es VRML (Virtual Reality Markup Language) [11], la cual provee las herramientas para el manejo de las características de comportamiento espacial de las figuras, con las cuales se les puede asociar movimientos y desplazamientos únicamente mecánicos. También provee comportamientos físicos para las figuras de manera tal que estas puedan ser cuerpos sólidos. Cualquier tipo de comportamiento conceptual debe ser construido desde un lenguaje externo al VRML. Como esta tecnología funciona sobre navegadores de Internet, los lenguajes con los que mejor interactúa son Java o JavaScript.

Resumiendo, los MOOs son mucho más apropiados para construir ambientes virtuales donde se requiera mayor calidad en el comportamiento conceptual, mientras que los ambientes de gráficos 3D son más apropiados para manejar comportamiento físico y espacial.

Este artículo presenta una nueva tecnología para la construcción de simulaciones de realidad virtual basados en EVOO (Entidades Virtuales Orientadas a Objetos), la cual modela una entidad virtual cualquiera que puede aparecer en un mundo virtual; y además, combina los conceptos de comportamientos físicos y espaciales de las tecnologías gráficas tridimensionales y el concepto de comportamiento conceptual al estilo MOO's. Las EVOOs están pensadas como un intermediario entre una aplicación real de software, dónde esta descrito el comportamiento propio del dominio de la aplicación, y una tecnología de gráfica tridimensional como interfaz (aunque esta asociación no es exclusiva, ya que también se le podría asociar a una interfaz de tipo de gráficos 2D o interfaz textual).

A partir de ahora y por el resto del artículo sólo hablaremos de la tecnología EVOO, por lo que cualquier referencia a los términos simulación o tecnología son con respecto a la tecnología de creación de realidades virtuales EVOO. También cualquier referencia al término comportamientos en plural, hace referencia a los comportamientos físicos, espaciales y conceptuales.

3 MODELO CONCEPTUAL DEL FRAMEWORK EVOO

La tecnología EVOO, simplifica las tareas de coordinación de las actividades de las distintas entidades dentro de la simulación. Esto se logra haciendo que toda entidad que conforma una simulación sea la suma de un conjunto específico de comportamientos espaciales, conceptuales y físicos. De esta manera se define el "comportamiento" de una entidad dentro de la simulación.

Para poder manejar los comportamientos como una única entidad, se diseñó una arquitectura que llamamos *Entidad Virtual.*

Una entidad virtual es la base sobre la que se agregan los comportamientos físicos, espaciales y conceptuales para particularizar el rol de cualquier entidad. De esta manera, todo lo que existe en una simulación es una entidad virtual.

Dentro de los ambientes virtuales encontramos la siguiente clasificación de entidades virtuales:

Habitante: es cualquier entidad virtual que existe dentro de un espacio virtual y que no tenga el rol de contener a otras entidades virtuales (una silla, un avatar, un robot, etc.). Por ser una entidad virtual de estas características, posee el comportamiento físico de tener un cuerpo o figura tridimensional sólida y el comportamiento espacial de poder desplazarse o ser desplazado dentro del espacio virtual.

Espacio virtual: es una entidad virtual con las características de comportamiento físico de un *contenedor* de habitantes, el cual no puede contener a otros espacios virtuales. Un espacio virtual no posee comportamiento espacial. Un espacio virtual es el lugar donde ocurren las acciones o interacciones entre los habitantes, donde las entidades tienen una representación física (es un espacio tridimensional por lo que usa un sistema de referencia física de tres ejes cartesianos para la ubicación de los habitantes). Un espacio virtual a priori es infinito y vacío, al igual que en las tecnologías 3D.

Avatar: es un habitante, el cual tiene el "comportamiento" de representar virtualmente a un usuario dentro del espacio virtual y es manipulado directamente por el mismo usuario. Esta entidad es la que le permite al usuario interactuar con su entorno. Un avatar posee las características de comportamiento físico y espacial de un habitante y además agrega características de comportamientos conceptuales, las que permiten el control por parte del usuario sobre el avatar. No existe ningún tipo de restricciones con respecto a la cantidad de avatars que pueden existir e interactuar en un mismo espacio virtual.

Conector: es un habitante que tiene el rol de conectar a dos o más espacios virtuales. Por cada espacio que conecte existirá una representación física del habitante Conector, pero todas las representaciones responderán como una sola entidad.

La Figura 1 resume esta clasificación de las entidades virtuales.

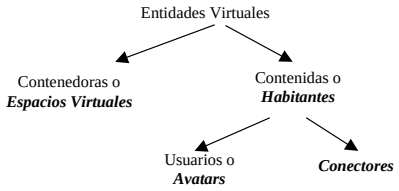

Figura 1. Jerarquía de especialización de las entidades virtuales

Además definimos:

- La metáfora *entorno*, como la porción de espacio virtual que rodea a una entidad virtual, dentro del cual puede alcanzar a otras entidades para interactuar. Por ejemplo, como sucede en la realidad una persona solo puede tomar las cosas que están a su alcance, si una cosa no estuviera al alcance de la persona, ésta debería desplazarse hasta que la cosa este a su alcance, es decir que el entorno es el alcance que tiene un avatar para poder interactuar.

- La metáfora *mundo virtual*, como a uno o más espacios virtuales conectados.
- Para que exista un ambiente virtual EVOO, debe existir al menos un espacio virtual.

4 ARQUITECTURA DE UNA ENTIDAD VIRTUAL

La arquitectura fue diseñada para dar soporte a los siguientes conceptos:

a) Una entidad virtual es la estructura básica para la creación de cualquier entidad dentro de una simulación construida con tecnología EVOO, la cual permite que la creación de cualquier entidad sea sencilla y fácil de particularizar.

b) Una entidad virtual es un conjunto de comportamientos espaciales, físicos y conceptuales, los cuales son asociados a una única unidad.

c) Proveer una manera consistente de comunicación con otras entidades virtuales, con otros objetos externos (clientes) y manejar la comunicación entre las tres clases de comportamientos de manera que estos puedan trabajar en forma conjunta.

Esta arquitectura fue construida sobre un ambiente de programación orientada a objetos, donde existe una clase llamada *Entidad Virtual* (EV) la cual modeliza las entidades virtuales como se ve en la Figura 1. Un *objeto cliente* es cualquier objeto externo (que no es instancia de la clase EV) que interactúe con un objeto de la clase EV.

Un objeto EV encapsula su comportamiento físico, conceptual y espacial. Los comportamientos son modelados por objetos (componentes de comportamiento), de manera tal que cada uno de los objetos se encarga de implementar su propio protocolo de comportamiento independientemente de los otros objetos comportamientos (estos se analizarán en más detalle en la sección 4.1).

Los objetos EV delegan todo el manejo de la comunicación de sus componentes internas de comportamiento a otro objeto llamado *FachadaInterna* (FI). Este objeto maneja los tres tipos distintos de comunicación que posee un objeto EV; estos son:

- Entre las componentes de comportamiento de un mismo objeto EV.
- Entre el objeto EV al que pertenece y otros objetos EV.
- Entre el objeto EV al que pertenece y un objeto cliente.
- Además, contienen las referencias de los objetos que comunica.

4.1 Comunicación entre las componentes de comportamiento de una EV y objetos clientes.

Un objeto de la clase EV puede interactuar con cualquier objeto cliente, por lo que la tecnología EVOO es totalmente transparente a otras aplicaciones orientadas a objetos.

Un objeto EV entiende los protocolos de sus componentes de comportamientos y los implementa como un *protocolo abstracto único*. Este diseño de protocolo único recibe un argumento, en el cual va encapsulado el mensaje de protocolo que alguno de las componentes de comportamientos que conforman al objeto EV entenderá. Los objetos clientes no tienen que implementar este diseño de protocolo único para poder interactuar con un objeto EV, ya que existe un objeto que denominamos AdaptadorClienteEV (AEV) el cual adapta el protocolo de mensajes para que estos puedan interactuar con un objeto EV sin necesidad de modificaciones. Un objeto AEV también adapta el protocolo para que el objeto EV interactúe con el objeto cliente.

La Figura 2 muestra el diseño UML [4] del objeto EV y sus colaboradores, *FachadaInterna*, *AdaptadorClienteEV*, *AdaptadorObjetoConceptual* y *EntidadFisicoEspacial*. Las flechas indican la relación *'conoce a'*. A través de los objetos *AdaptadorObjetoConceptual* (AOC) y

EntidadFisicoEspacial (EFE) es que el objeto EV se relaciona con sus componentes de comportamientos.

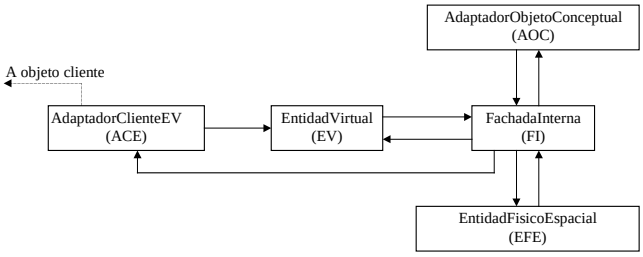

Figura 2. Diseño de una entidad virtual con sus objetos colaboradores

Específicamente a través del objeto AOC el objeto EV se relaciona con su componente de comportamiento conceptual, el cual será una subclase del objeto OC (*ObjetoConceptual*) y es en donde el desarrollador definirá el comportamiento conceptual de la entidad virtual. El objeto AOC está subclasificado en dos clases las cuales permiten definir si la entidad es un habitante Conector o no, estas clases se denominan *MultiplesEV* y *UnicoEV* respectivamente.

Por otro lado el objeto EV se relaciona con sus componentes de comportamientos físicas y espaciales a través del objeto EFE. El objeto EFE esta subclasificado en dos clases: *EntidadContenedor* y *EntidadHabitante*, en estas subclases se define si la entidad virtual es un espacio virtual o un habitante. Dentro de las subclases se encuentran la estructura que contiene a las componentes de comportamientos físicas y espaciales de la entidad virtual, las cuales son las que particularizan los desarrolladores.

4.2 Comunicación entre EVs

Como se mencionó anteriormente, el objeto EV es principalmente un diseño de comunicación, donde el objeto FachadaInterna es el corazón de este diseño. El diseño del objeto FI responde al Pattern Façade cuyo propósito es: "*Proveer una interfaz unificada a un conjunto de interfaces dentro de un subsistema. Façade define una interfaz de alto nivel que hace al subsistema fácil de usar*"[6]. En nuestro caso particular los subsistemas son las componentes de comportamiento de un objeto EV. El objeto FI hace que la comunicación sea entre objetos EV y no entre las componentes de comportamientos de distintos objetos EV.

El funcionamiento de la comunicación en un objeto EV es el siguiente:

Para mensajes de entrada: los mensajes los recibe el objeto EV, el cual los delega a su objeto FI para que este determine cual de los objetos comportamientos es el que responde al mensaje.

Para mensajes de salida: Los mensajes son generados por las componentes de comportamientos, las cuales le dicen al objeto FI cual es el objeto destinatario y cual es el mensaje, el objeto FI lo delega al objeto EV y este efectúa el envío del mensaje.

4.3 Manipulación de Referencias

Un objeto FI contiene y administra un conjunto de referencias a otros objetos, el cual permite hacer un manejo dinámico de las referencias, de manera tal que las componentes de comportamientos puedan hacer referencia a un mismo objeto FI y guardar en él sus referencias. Un objeto FI permite a cualquiera de los objetos que lo conoce agregar y acceder a una referencia libremente, pero establece un control de eliminación de referencias que se expondrá más abajo.

Los objetos FI delegan la contención de éstas referencias a un objeto colaborador llamado *Colector*, el cual guarda las referencias asociadas a un identificador. Cuando se agrega una referencia se establece cual será su identificador asociado, de esta manera cualquier objeto puede pedir acceder a ésta referencia invocando el identificador correspondiente.

El objeto FI contiene normalmente las referencias a las componentes de comportamientos y al objeto *AdaptadorClienteEV* por parte del objeto EV y las referencias a otros objetos EV por parte de los objetos colaboradores.

El objeto FI al contener las referencias de más de un objeto muy probablemente se encuentre con casos de duplicidad de referencias. Para resolver este problema se creo una jerarquía de clases de referencias en la cual se implementan los tres tipos de referencias que se necesitan para manejar la comunicación entre las componentes de comportamientos, estas referencias las denominamos: estática, dinámica y monitoreada.

*Referencia Estática***:** son aquellas que no pueden ser modificadas ni borradas del conjunto de referencias por ninguno de las componentes de comportamientos. De manera tal que permanezcan estáticas desde su creación. Estas sólo pueden ser accedidas, como es el caso de las referencias a las componentes de comportamientos, al objeto EV y al objeto *AdaptadorClienteEV*.

La referencia Dinámica: son aquellas que las componentes de comportamientos pueden modificar, pero no borrar del conjunto de referencias. Este es el caso de un objeto EV clasificado como habitante, el cual debe conocer al objeto EV clasificado como espacio virtual que lo contiene, la referencia a este último objeto debe ser dinámica, ya que la entidad virtual habitante podría ser desplazada a otro espacio virtual.

Referencia Monitoreada: puede ser compartida pero no modificada y sólo el objeto colaborador que la crea puede borrarla. Este tipo de referencias, por ejemplo, es la que utiliza el objeto conceptual de un objeto EV clasificado como avatar, para conocer los objetos EV clasificados como habitantes que están dentro de su entorno.

5 COMPONENTES DE COMPORTAMIENTO EN UNA EV

En esta sección se analizarán por separado cada una de las componentes de comportamientos, cuales son sus estructuras básicas y cuales sus metodologías de particularización.

5.1 Comportamiento Conceptual

En lo que sigue de esta sección nos enfocaremos en el modelo de las componentes de comportamientos conceptuales, la cual según la definición conceptual de la clasificación de comportamientos: *‘es todo aquel comportamiento que no es ni físico ni espacial, que se le puede agregar a una entidad para particularizarla o darle cierto grado de “inteligencia”’*. Las componentes de comportamientos conceptuales están modeladas por las subclases de la clase abstracta ObjetoConceptual (OC).

5.1.1 *Objetos conceptuales*

Las componentes que modelan los comportamientos conceptuales son definidas por los desarrolladores y deberá existir uno por cada objeto EV que exista dentro de una simulación. Existe una subclase definida especialmente la cual implementa el caso en que un objeto EV no necesite tener ningún comportamiento conceptual definido, ésta subclase del objeto OC la denominamos OCNulo, Una componente de comportamiento conceptual puede manejar a varios objetos EV a la vez, pero la componente en sí no contendrá las referencias de él o los objetos EV a los que maneja. La clase OC contiene una referencia a un objeto que es el encargado de contener las referencias a él

o los objetos EV a los cuales las subclases de OC manejarán. Este objeto, lo denominamos AdaptadorObjetoConceptual (AOC), de este objeto sé hablar en la siguiente sección.

Las únicas reglas que se deben respetar dentro de cualquier subclase de la clase OC son:

1. Que utilice el formato de protocolo único para comunicarse tanto con las otras componentes de comportamientos como con un objeto cliente y otros objetos EV.
2. Que ninguna subclase de OC contendrá en sí misma la o las referencias a los objetos EV que maneje, esta función es delegada al objeto AOC asociado y accederá a los objetos EV a través de identificadores.

El desarrollador no posee mayores restricciones que las expresadas anteriormente. Normalmente dentro del objeto conceptual se definen cosas del estilo de disparadores de eventos a las otras componentes de comportamientos los cuales están asociados a algún evento no mecánico, para el almacenamiento de información relevante al funcionamiento del objeto EV, también para hacer algún tipo de análisis lógico el cual determine una forma particular de comportamiento, para agregar o quitar referencia a objetos EV del objeto FI.

Como mencionamos más arriba, una componente de comportamiento conceptual puede controlar a más de un objeto EV, esto permite modelar situaciones en la que más de un objeto EV clasificado como habitante, los cuales pueden estar en distintos espacios o no, deban responder a un mismo objeto conceptual controlador, este tipo de situación es el presentado por los objetos EV clasificados como Conectores.

5.1.2 *Adaptadores objetos conceptuales*

El objeto AdaptadorObjetoConceptual (AOC) es sólo un objeto abstracto del cual cuelgan las subclases instanciables UnicoEV y MultiplesEV, estas subclases implementan los dos casos posibles de contención de referencias que puede necesitar una componente de comportamiento conceptual, el caso de que la componente maneje a un único objeto EV o el caso de que maneje múltiples objetos EV. Los objetos subclases de AOC en realidad conocen a objetos FI y no a objetos EV.

Un objeto de la subclase de AOC es uno de los objetos colaboradores que conoce el objeto FI, a través del cual se asocia al objeto EV un objeto subclase de OC.

Los objetos subclases de OC no están preparados para recibir mensajes en el formato del protocolo único, por lo que cualquiera de los dos objetos subclase de AOC hace la transformación para que el mensaje pueda ser interpretado.

En lo que sigue de la sección analizaremos las componentes de comportamientos físicos y espaciales de un objeto EV, se mostrará como su funcionamiento es totalmente independiente, pero como su diseño esta pensado para que colaboren estrechamente.

Es muy importante aclarar que ambos comportamientos sólo manejan y contienen la información necesaria para poder tener una representación actualizada de la figura que existe dentro de una simulación generada por una tecnología gráfica.

5.2 Comportamiento Físico

Una componente de comportamiento físico esta representado por una figura. Conceptualmente una figura puede ser simple o compuesta. Para modelar este concepto se diseño una jerarquía de objetos figuras, las cuales tienen a la cabeza a la clase abstracta Figura y como subclases a las clases instanciables Simple y Compuesta.

Una figura simple va asociada a un objeto que modela una geometría tridimensional básica. Los objetos que modelan los conceptos de geometrías tridimensionales básicas los denominamos

Formas. Existen objetos predefinidos como geometrías básicas los cuales son subclases instanciables del objeto abstracto Forma, estos objetos son Cajas, Esferas, Pirámides, Conos y Cilindros. Estos objetos subclases de Forma son las primitivas de diseño Físico de la entidad virtual.

Una figura compuesta, conceptualmente, es una agrupación de más de una figura simple o compuestas Los objetos de la clase Compuesta hacen que su agrupación de figuras se comporte como un sólo cuerpo. Los objetos la clase Compuesta pueden agrupar dentro de sí a objetos de la clase Simple, a objetos de la clase *ControladorEspacial* o a ambas clases de objetos mezclados. Los objetos de la clase *ControladorEspacial* se verá en la siguiente sección.

Cualquiera de las subclases del objeto Figura conoce: su punto de origen tridimensional, su grado de inclinación, la distancia de desplazamiento del punto de origen y el identificador de la figura.

Estos datos de información acerca de la figura en el espacio, son los que permitirán a las componentes de comportamientos espaciales controlar los movimientos de un objeto subclase de Figura.

Los desarrolladores deberán crear y componer las figuras manualmente.

5.3 Comportamiento Espacial

La última componente de comportamiento que veremos es la espacial, la cual por la definición conceptual de la clasificación de los comportamientos, vista al comienzo de este artículo, es: *"aquellos que permiten a una entidad moverse, desplazarse o ser desplazada dentro de un espacio virtual"*. Este comportamiento es solo un atributo de la entidad a la cual pertenece y particulariza, por lo que su diseño esta muy vinculado a la componente física de una entidad.

Las componentes de comportamiento espacial encierran a una componente de comportamiento físico para darle a este último componente características de comportamientos espacial. Para representar esto, se diseño un objeto que encapsule a la componente de comportamiento físico, este objeto lo denominamos *ControladorEspacial*.

Por un lado del diseño, un objeto de la clase *ControladorEspacial* encapsula en sí mismo a un objeto Simple o Compuesto y por otro tiene la estructura para agregar comportamientos espaciales. Estos comportamientos espaciales están modelados por objetos y están clasificado en dos tipos comportamientos: los predefinidos y los particularizables. Los objetos que modelan los comportamientos espaciales predefinidos establecen dos subtipos, los comportamientos que hacen que una figura rote, y los comportamientos que hacen que la figura sea desplazada. Para implementar estos subtipos de comportamientos espaciales predefinidos se creó una jerarquía en la cual en el tope se define la clase abstracta *CEP* (siglas de *Comportamiento Espacial Predefinido*), de donde cuelgan las subclases *Rotador* y *Desplazador*. Estos objetos que modelan los comportamientos espaciales predefinidos son solo las herramientas (métodos) mínimas básicas para poder controlar a los objetos subclases de Figura.

Los *comportamientos espaciales particularizables* son implementados por las subclases de la clase abstracta *ManejadorHabitante*, estas subclases deberán ser creadas por los desarrolladores para darle a un objeto de las subclases de Figura algún tipo comportamiento espacial. En ella se escribirán los métodos específicos que permitirán particularizar los movimientos que tendrá un objeto de subclase de Figura. Para poder manejar a la figura los objetos subclases de *ManejadorHabitante* se valen de los métodos (herramientas) que están definidos en los objetos *Rotador* y *Desplazador*.

Para que un desarrollador pueda agregar comportamientos espaciales a un objeto simple o compuesto deberá crear un objeto *ControladorEspacial*, al cual le dará a conocer a que objeto subclase de Figura controla, y que tipo de comportamientos predefinidos necesitará para poder

particularizar el comportamiento espacial, y cual es el objeto subclase de ManejadorHabitante tiene como manejador.

En el final de la sección analizaremos los conceptos que define la clasificación de entidad Contenedor y entidad Habitante, vistos en la sección Tecnología EVOO. Estos conceptos están modelados por las subclases instanciables de la clase abstracta EntidadFisicaEspacial (EFE).

5.3.1 *Entidades físicas espaciales*

Las subclases de *EFE* son denominadas *EntidadContenedor* y *EntidadHabitante*. La clase *EntidadHabitante* solo contiene un objeto subclase de Figura o un objeto *ControladorEspacial*, clases de las cuales ya hemos analizado. La clase *EntidadContenedor* modela los conceptos de espacio virtual vistos en la sección 3. Un objeto *EntidadContenedor* permite agregar o eliminar objetos EV, también permite interactuar directamente como los objetos EV que contiene. Para poder ser particularizado un objeto de la clase *EntidadContenedor* se debe crear objetos los cuales son subclases instanciables de la clase abstracta *ManejadorContenedor*, en estas subclases se implementan todas las características de comportamiento que necesite el objeto *EntidadContenedor*. Estás subclases al igual que las de la clase ManejadorHabitante son creadas e implementadas por los desarrolladores.

6 CASO DE ESTUDIO

Cuando se crea una instancia de un objeto EV, esta instancia tiene que ser particularizada como *Contenedor*, *Habitante*, *Conector* o *Avatar*. Cada una de estas instancias particularizadas tiene diferencias en su construcción (se recomienda ver el gráfico de sección 2), estas diferencias están dadas por las subclasificación de los objetos *AOC* y *EFE*, las cuales son como sigue a abajo:

- Un objeto EV instanciado como *Contenedor* conoce a una instancia del objeto *EntidadContenedor* como *EFE* y una instancia del objeto *UnicoEV* como *AOC*.
- Un objeto EV instanciado como *Habitante* conoce a una instancia del objeto *EntidadHabitante* como *EFE* y una instancia del objeto *UnicoEV* como *AOC*.
- Un objeto EV instanciado como *Avatar* conoce a una instancia del objeto *EntidadHabitante* como *EFE* y una instancia del objeto *UnicoEV* como *AOC*.
- Un objeto EV instanciado como *Conector* conoce a una instancia del objeto *EntidadHabitante* como *EFE* y una instancia del objeto *MultipleEV* como *AOC*.

Estas instancias del objeto EV son las que determinan las primitivas de diseño de la tecnología EVOO, y a través de ellas (particularizadas) se crea cualquier entidad dentro de una simulación. Para poder particularizar cualquier entidad el desarrollador deberá agregarle a la instancia del objeto EV ya creado los siguientes objetos según sea el tipo de instancia. Los objetos a agregar son:

- Subclase de la clase *ObjetoConceptual*, para especificar comportamiento conceptual.
- Subclase de la clase *ManejadorHabitante*, para especificar comportamiento espacial a una entidad clasificada como habitante.
- Subclase de la clase *ManejadorContenedor*, para especificar comportamiento físico en una entidad clasificada como contenedor.
- Composición de instancias de objetos subclase de Figura, para especificar comportamiento físico a entidades clasificadas como habitante.

7 CONCLUSIONES

En este artículo se presentó un framework para la construcción de ambientes virtuales basado en Entidades Virtuales. Principalmente las Entidades Virtuales se caracterizan por tener tres tipos de

comportamientos básicos: físico, espacial y conceptual, lo que permite reunir en una misma componente los distintos aspectos de una entidad de un mundo virtual.

Este framework se ha desarrollado en un entorno Smalltalk, respetando las descripciones enunciadas en este artículo. Por el momento, este framework permite conectar cualquier aplicación orientada a objetos con los EVOO a través del comportamiento conceptual. Del otro lado de las interfaces, una EVOO puede ser conectada con cualquier interfaz 3D, con solo implementar las conexiones necesarias. En particular, en la actualidad se está de diseñando la manera de conectar EVOO con VRML.

8 REFERENCIAS

[1] Benford, S. & Fahlen, L. (1993) *A Spatial Model of Interaction in Virtual Environments*. In Proc. Third European Conference on Computer Supported Cooperative Work (ECSCW'93), Milano, Italy.

[2] S. Benford, C. Greenhalgh, D. Lloyd. *Crowded Collaborative Virtual Environments*, Proc. 1997 ACM Conference on Human Factors in Computing Systems (CHI'97), pp. 59-66, Atlanta, Georgia, US, March 22-27, 1997.

[3] S Benford, J. Bowers, L. Fahlen, C. Greenhalgh, D. Snowdon (1995*) User Embodiment in Collaborative Virtual Environments*. In Proc. 1995 ACM Conference on Human Factors in Computing Systems (CHI'95), May 7-11, Denver, Colorado, USA, ACM Press.

[4] G.Booch, I. Jacobson, J. Rumbaugh *The Unified Modeling Language User Guide* (The Addison-Wesley Object Technology Series). Addison-Wesley Pub Co; ISBN: 0201571684. October 1998.

[5] *E_MOO*: Web site dedicated to MOO and MUD research. http://tecfa.unige.ch:4243/

[6] E. Gamma, R. Helm, R. Johnson, J. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison Wesley. October 1994.

[7] C. Greenhalgh, *Large Scale Collaborative Virtual Environment*. Distinguished Dissertation, Springer Verlag, 1999

[8] C. Haynes and J. Rune Holmevik Eds. *High Wired: On the Design, Use, and Theory of Educational MOOs*, University of Michigan Press.

[9] S. Harrison, and P. Dourish. *Re-Place-ing Space: The Roles of Places and Spaces in Collaborative Systems*. In Proc. ACM 1996 Conference on Computer Supported Cooperative Work (CSWC'96), November 16-20, Massachusetts, USA. ACM Press.

[10] *LambdaMOO: One of the oldest, most diverse, and largest MOOs in existence*. Telnet: telnet://lambda.moo.mud.org:8888

[11] Andrea L Ames, David R. Nadeau and Jhon L. Moreland, *VRML 2.0 Sourcebook*. San Diego Supercomputer Center.