Aspectos clave en las tecnologías de Mundos Virtuales Distribuidos (DVE)

Pedro Morillo♦, Marcos Fernández♦, Juan Manuel Orduña♦

Resumen— **Dos factores clave han facilitado el auge de las nuevas tecnologías basadas en mundos virtuales distribuidos. Estos han sido el desarrollo de tarjetas gráficas 3D de altas prestaciones y el incremento del ancho de banda de las líneas telefónicas domésticas para la conexión a Internet.**

El presente trabajo muestra en profundidad el funcionamiento básico de este tipo de arquitecturas, así como las ventajas y principales problemas tecnológicos que suelen presentar el diseño e implementación de las aplicaciones basadas en este paradigma.

Uno de estos puntos clave es el problema del particionado. El trabajo recoge una nueva solución para éste, basada en técnicas de programación evolutiva.

Los resultados obtenidos, comparados con la principal referencia en este área, demuestran que se obtienen mejores soluciones en la reconfiguración del mundo virtual empleando para ello menores tiempos de ejecución.

Palabras Clave—**Mundos Virtuales Distribuidos, taxonomía DVE, problema particionado, avatar, equilibrado de cargas.**

I. INTRODUCCIÓN

Una gran parte de las tecnología empleadas en el campo de los gráficos por computador han evolucionado a un actual modelo de desarrollo técnico basado en Realidad Virtual (RV). Este término ha sido profusamente definido en la literatura de materia por numerosos investigadores, pero ha sido Roy Stuart del NYNEX quien en [30] puntualiza la RV como un entorno sintético en tres dimensiones interactivo, inmersivo y multisensorial. La RV proporciona la base para el desarrollo de un amplio rango de aplicaciones, desde campos tales como el mundo del entretenimiento, la educación, formación civil, hasta el tratamiento y representación de información en medicina, robótica, química, y una gran multitud de otros propósitos.

Particularmente hablando, los agentes inteligentes en este tipo de entornos, pertenecen a las denominadas tecnologías de "espacio compartido". Estas tecnologías, tal y como muestra la figura 1, fueron clasificadas por Benford et al en [5] creando una relación, hasta la actualidad poco comentada, entre la localización o modo de transporte de los elementos de una aplicación y su grado de artificialidad o presencia en ella.

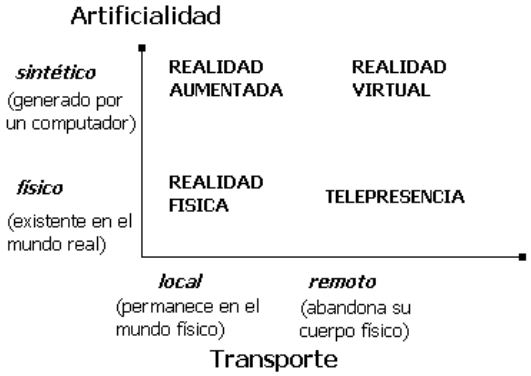


Fig. 1. Tecnologías espacio-compartido según Benford

El eje de transporte indica el nivel en que los usuarios o participantes van a abandonar su localidad espacial, y el de artificialidad el grado en que un ambiente representado es generado por un computador. Empleando esta bi-dimensionalidad cuatro estados pueden ser clasificados: Realidad física, que reside en el mundo físico, real y local donde los objetos son tangibles y los participantes corpóreos. A la inversa, la realidad virtual permite que los participantes se introduzcan de forma inmersiva en un mundo sintético y remoto.

En la telepresencia, los usuarios están presentes en un mundo real localizado remotamente a su posición física en el mundo (ej. operaciones con robots en Marte dirigidos desde la tierra).

En la realidad aumentada, objetos sintéticos, obtenidos a través del procesamiento de un computador, son añadidos a un ambiente local físico y real.

♦ Grupo ARTEC. Instituto de Robótica. Universidad de Valencia. Correo Electrónico: {Pedro.Morillo,Marcos.Fernandez}@uv.es.

Así pues, con esta estructura, en la categoría de "realidad virtual" se agrupan una gran disparidad de sistemas interactivos distribuidos, para los cuales cuatro grandes focos de investigación pueden diferenciarse :

- ✓ Simulaciones civiles ([3], [32] y [33]) y militares ([25] y [26]).
- ✓ Realidad virtual (VR) asociada al modelado y dinámica individual de escenarios en un computador. ([30]).
- ✓ Trabajo colaborativo apoyado en computación (CSCW). Ampliamente descritos en [23] y [28].
- ✓ Videojuegos en red multijugador para la industria del entretenimiento. ([1] y [17])

II. EXTENSIÓN DEL CONCEPTO DE RV A DIS, DVE Y CVE

Desde 1980, el Ministerio de Defensa de los Estado Unidos de América lleva desarrollando aplicaciones militares empleando un protocolo para la simulación interactiva denominado DIS. Este protocolo, debido a su empleo extensivo, fue reconvertido a un standard de jure IEEE en 1992 (tal y como se observa en [21] y [26]) y ha evolucionado en una arquitectura de alto nivel para el desarrollo de aplicaciones distribuidas denominada HLA. HLA se fundamenta en proporcionar al programador un mecanismo de intercambio de datos, basado en federaciones, mediante el cual crea un arquitectura basada en la distribución de servicios, con la característica que estos no tienen por que estar orientados a propósitos militares, a diferencia de DIS ligado intrínsecamente a ellos. Fruto de esta evolución, en la investigación de sistemas cooperativos para el diseño de aplicaciones, han sido los acuerdo entre militares e industrias del entretenimiento estadounidenses para la creación de una línea de investigación en el desarrollo de videojuegos multijugador, de alta calidad visual, con usuarios remotos.

De forma paralela a estos sistemas, nacidos en el seno militar, que basan su funcionamiento en simulaciones con un gran número de usuarios en la escena, desde el principio de los 90 una nueva rama de la Realidad Virtual enfoca sus investigaciones en los DVE o ambientes virtuales distribuidos. Varios proyectos nacen dentro de esta iniciativa tales como DIVE ([9]), GreenSpace ([22]), Spline ([4]), Community Place ([16]) y Massive ([12]), caracterizados por un diseño orientado al uso local, para sistemas multiplataforma a bajo coste y soporte para un reducido número de participantes.

A su vez, otros equipos de investigación de diferentes universidades a nivel mundial, durante la pasada década realizan el esfuerzo en la combinación de los CSCW con la Realidad Virtual dando lugar a los ambientes virtuales colaborativos o CVE, descritos en [23] y [28]. Estos se diferencian de los anteriores en el fuerte enfoque que estos realizan sobre la colaboración de los avatares, es decir, los usuarios del sistema que al mismo tiempo desde emplazamientos distintos están, sobre el ambiente virtual, compartiendo el mismo objeto.

Para conseguir estos objetivos, desde cualquiera de sus perspectivas, es necesario destacar e identificar cuales son los cuatro aspectos más relevante a nivel de red que van a limitar cualquier aplicación que se base en este modelo:

A. Recursos inherentes a la red

Toda simulación distribuida se enfrenta a tres claras limitaciones de recursos:

Ancho de banda: Referida a la capacidad de comunicación que va a tener la linea de la red a través de la cual los datos de la aplicación van a ser transmitidos. Visto de otro modo, es la cantidad de datos emitidos o recibidos por unidad de tiempo. Dado que no es lo mismo que los usuarios del sistema se comuniquen a través de una red de área local (LAN) o realicen accesos remotos con una WAN, la frecuencia de los mensajes de los agentes, así como el tamaño de estos ha de ser factor muy importante en el diseño del trafico de red.

Latencia: Indica la longitud de tiempo (o retraso) que transcurre desde que un elemento o agente de la red envía un mensaje hasta que su destinatario lo recibe. Por definición la latencia en un sistema en red nunca puede ser eliminada y junto a ella aparecen fenómenos que pueden todavía agravar más sus consecuencias. Tal es el jitter, o la varianza de la latencia entre dos punto distintos de la red, que va a hacer que en muchas ocasiones la sincronización de diferentes agentes del sistema no se haga todo lo exacto como se deseara.

En sistemas interactivos en tiempo real multiusuario se considera aceptable una latencia máxima entre 0.1 y 1.0 segundos. A pesar de ello, el citado estándar DIS especificaba una latencia máxima de red no superior a 100ms y en sistemas

RTS o estrategia tiempo real se permiten retrasos de hasta 500ms.

Potencia de cálculo: El tratamiento del tráfico de red supone un coste computacional extra a cualquier equipo conectado a ella dentro del sistema distribuido. En determinadas ocasiones esta fuente de información puede incluso convertirse en una fuente de problemas debido al tiempo que el computador va a emplear para solicitar y responder peticiones. Empleando estimaciones conservadoras, Singhal en [29], demuestra que para una simulación de 100.000 entidades sobre un sistema distribuido en red se requiere el 80% de la potencia de cálculo de CPU en un equipo con procesador a Pentium II a 500Mhz.

B. Redistribución de elementos

Dado que no se pueden realizar mejoras basándose en unos limitados recursos de red, la optimización de soluciones, para los problemas de sistemas interactivos distribuidos, debe de ser tratada desde un punto de vista de más alto nivel. Esto conlleva la elección de nuevas arquitecturas para comunicación, datos y control.

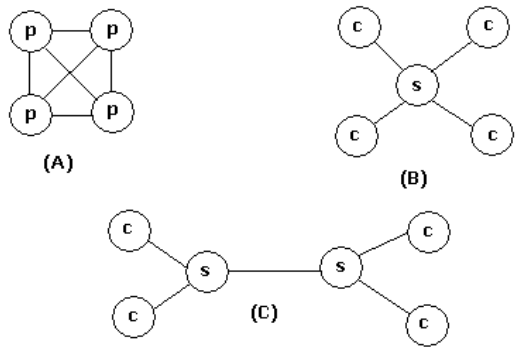


Fig.2. Clasificación arquitecturas de comunicación

La arquitectura de comunicación (fig2) debe de ser elegida entre diferentes modelos, a destacar, arquitecturas punto-a-punto, donde se tiene todo nodos conectado en la red contra todos ellos (A), cliente-servidor, donde toda la comunicación es controlada por uno nodo servidor mientras que el resto tienen el papel de cliente de este (B), y arquitecturas red-servidor caracterizadas por tener un conjunto interconectado de servidores punto-a-punto entre ellos y que se comunican de forma cliente-servidor entre sus subredes (C).

La actual linea de investigación en DVE's está últimamente realizando un uso muy extensivo de estas dos últimas arquitecturas para la implementación de sistemas. Así es posible encontrar en los textos científicos nomenclaturas del estilo de SSDVE (Single Server Distributed Virtual Environment) en [lui1] para designar a las arquitecturas cliente-servidor. Para el caso de las arquitecturas red-servidor la variedad es todavía mas amplia. Así, en la serie anterior de artículos Lui-Chan ([18], [19] y [20]) aparece el termino MSDVE (Multiple Server Distributed Virtual Environment) para referenciarlos. A su vez en [10] se les denominan topologías jerarquicas o en [11] arquitecturas servidor-servidor.

Las arquitecturas de control y datos hacen referencia a dos propiedades muy importantes para los sistemas en red que son consistencia y capacidad de reacción. Para lograr una alta consistencia, la arquitectura debe garantizar que, procesos ejecutados en nodos remotos no dependan del nodo donde sean alojados, de forma que la visión que cada uno de los agentes tenga del resto de ellos sea independiente a la topología de la red, tal y como si estuvieran corriendo en la misma máquina. Proporcionar mejor capacidad de reacción implica disminuir los tiempos de respuesta media ante cualquier petición o consulta de los agentes del sistema, lo cual va a conllevar mas computación para reducir los requisitos de latencia y ancho de banda de la aplicación.

En los DVE's son tan importantes las arquitecturas de datos como las arquitecturas de comunicación de los elementos del sistema. En 1997 M.Macedonia define en [21] una buena aproximación a la caracterización del modelo de datos para los entornos virtuales distribuidos.

En ésta, el autor define cuatro metodologías diferentes para la distribución de los datos en la que podemos encontrar:

- Mundos homogéneos replicados. Un método común en grandes DVE's es inicializar el estado de cada participante en la escena con una base de datos homogénea que contiene información sobre el terreno, geometría de los modelos, texturas y modelos de comportamiento para cada elemento representado en el mundo virtual. La comunicación entre todos y cada uno de los miembros de la escena se va a basar, o bien, en el cambio del estado de determinados objetos tales como la posición de un vehículo móvil, o bien en la producción de eventos como la colisión de misiles simulados o la colisión entre dos objetos. La ventaja de esta aproximación es que los mensajes son relativamente pequeños. Por el contrario, este modelo es relativamente inflexible ya que por

una parte todo elemento de la simulación ha de recoger modificaciones de objetos que incluso no le atañen y además las posibles perdidas de estos objetos a largo plazo, puede en numerosas ocasiones, causar problemas de inconsistencia entre los participantes del mundo virtual. Éste es el modelo típico del famoso DVE SIMNET recogido en [25] y comentado mas adelante, donde, una vez la simulación comienza, cada uno de los elementos que la compone se guarda una copia local de la base de datos inicial de ésta, despreocupándose de garantizar la consistencia excepto con mensajes de detección de elementos vivos y algún evento.

- Mundos centralizados y compartidos. Es el esquema de los clásicos mundos MUD, descritos en [7], donde existe una única base de datos del mundo compartida por todos los participantes en la simulación. La limitación de estos mundos es clara con la imposibilidad de crear simulaciones de más de una decena de participantes y con mundos asociados muy sencillos.

- Mundos distribuidos con actualizaciones punto-a-punto. Metodología nacida como similitud a las arquitecturas hardware de memoria compartida. Se caracterizan por estar compuestos por participantes independientes, que ejecutan procesos de forma paralela, representando elementos en el mundo virtual. La escena se encuentra dividida en dos grandes componentes. La primera de ellas corresponde a los modelos estáticos del sistema. Estos modelos se caracterizan por ser invariantes en propiedades durante el transcurso de la simulación y residen de forma local, y replicada, en todos los nodos del sistema. Al contrario de estos, se encuentran los modelos dinámicos de la simulación que se encuentran esparcidos entre los diferentes participantes. La localización de cada uno de estos elementos es transparente a los usuarios, que ven todo el mundo virtual como residente en su propia máquina, gracias a la ilusión de proximidad que aporta el gestor de mundos.
  Estos modelos plantean problemas de escalabilidad, tanto del número de participantes en la simulación como del tamaño del mundo virtual, debidos a los costes de comunicación asociados con el mantenimiento de la consistencia y fiabilidad del mundo. A pesar de ello, tal y como se comenta en [9] para la implementación del DVE DIVE, este paradigma está claramente recomendado en simulaciones donde la carga estática del modelo es grande, tales como vuelos sobre terrenos virtuales, juegos basados en plataformas o recorridos sobre decorados interiores.

- Mundos distribuidos con actualizaciones cliente-servidor. La última de las arquitecturas hace referencia a una variación del modelo general de aplicaciones cliente-servidor. En ella, el modelo global del mundo es repartido entre los participantes en la simulación cuyas comunicaciones son reguladas por un servidor central. Usualmente, en estos modelos, tal y como ocurre en el DVE Massive de la Universidad de Nottingham, descrito en [12], cuando una entidad de un cliente cualquiera se desplaza a través del mundo virtual su base de datos es constantemente actualizada por un gestor central de objetos. Este gestor tiene conocimiento total de que cliente es el que actualmente mantiene la parte del mundo sobre la cual se está desplazando esa entidad.
  A pesar de que estas simulaciones permiten la existencia de varios servidores como gestores de las comunicaciones, éstos se convierten rápidamente en cuellos de botella para la E/S incrementando la latencia, ya de por si inherente, de los DVE.

C. Escalabilidad

Por definición, va a ser la capacidad mostrada por el sistema para adaptarse a cualquier cambio en los recursos del mismo. Para nuestro sistema o arquitectura, esto va a verse traducido en cuales van a ser los mecanismos de respuesta que este dispondrá para ajustarse a una rápida variación del número de agentes en un momento dado, o como va a poder ser blindado el sistema para que una reducción del ancho de banda del canal de comunicación, en un momento dado, le afecte lo menos posible.
Para lograr esta clase de escalabilidad, se debe de dotar al sistema de dispositivos físicos adicionales (soluciones basadas en el hardware) o añadir nuevos agentes (software) que de forma paralela a los lanzados por los usuarios de la aplicación permitan al sistema concurrencia lógica de computación.

D. Seguridad y engaño

La seguridad “on-line” se ha convertido en uno de los grandes puntos a resolver por cualquier tipo

de aplicación que trabaje sobre la red, no sean solo aplicaciones distribuidas, sino servidores, bases de datos o sistemas de consulta y almacenamiento masivo.
A pesar de que son publicados miles de informes sobre agujeros de seguridad on-line en la mayoría del software comercial existente en la actualidad, protocolos como DIS, anteriormente citado, no incluyen entre sus objetivos defensa para accesos no deseados, tratamiento de información cifrada o protección de datos para usuarios.

Por ello, para este tipo de sistema es necesario cubrir dos puntos de vista muy importantes a este nivel:

- ✓ Información sensible a su protección (tal y como número de tarjetas de crédito, contraseñas de acceso,...)
- ✓ Proporcionar un campo de ejecución justo, donde no hay lugar al autoengaño y se asume que el engaño es tan difícil como posible.

III. PROBLEMA PARTICIONADO

A pesar de la heterogeneidad del modelo descrito anteriormente, existe un conjunto de características comunes a los modernos DVEs. Se pueden encontrar numerosas referencias en la literatura de la materia, ([29][21] y [18]) en las cuales la implementación de los DVE's sobre arquitecturas red-servidor se está convirtiendo en un standard de facto. Tecnologías de red basadas en el procesamiento paralelo, como la gestión de servidores web de alto rendimiento, están llegando a este nivel a las mismas conclusiones ([13]).
En estas arquitecturas, denominadas también jerárquicas por algunos investigadores ([3]), la simulación es controlada por un conjunto de servidores interconectados. Cuando un cliente desea introducirse en una simulación éste obligatoriamente es asignado, a uno y sólo uno, de estos servidores coordinados. Así, para cada actualización (ej.: cambio de posición) de su avatar asociado, el cliente envía un mensaje al servidor, el cual, atendiendo a la configuración del sistema en ese momento, redirige la comunicación hacia otro servidor de la simulación o, si es el caso, a sus propios clientes.
Para evitar un desbordamiento del volumen de mensajes circulando en la red se suelen definir áreas de influencia (AOI en [29], locales en [12] y aura en [32]) para los avatares. Esta propiedad permite que un avatar sólo propague mensajes, hacia otros avatares que pertenezcan a su AOI (del inglés "area of influence").
Dependiendo de la localización del receptor de un mensaje pueden definirse en un DVE (fig.1) dos esquemas diferentes de comunicación. Si éste es alojado en el mismo servidor que el emisor se realizarán rápidas transferencia intra-servidor; en caso contrario, serán necesarias costosas comunicaciones inter-servidor para que dos avatares puedan intercambiar información.

Fig. 1. Modelo comunicaciones básico en un DVE

Sobre esta arquitectura, Lui y Chan en [18] redefinen el *problema del particionado.* Este consiste en la búsqueda de la mejor asignación, del conjunto de clientes a los servidores de la simulación, que permita una óptima interactividad a los usuarios de aplicaciones (a nivel de *frame-rate*) mediante la minimización del tráfico de red. Se han de considerar dos factores básicos a la hora de crear una eficiente asignación. En primer lugar, C_p^W , el número de avatares asignados a cada servidor ha de ser lo más similar posible, creando el denominado balanceo o equilibrado de cargas. En segundo lugar, para minimizar el volumen total de comunicaciones inter-servidor, C_p^L, avatares localizados en emplazamientos vecinos deberían de ser asignados al mismo servidor. Dado que simultáneamente para todos los avatares de la escena esto no es posible, es entonces necesario encontrar un adecuado agrupamiento topológico. De acuerdo a estos dos parámetros, se ofrece una función de evaluación del coste para estimar la bondad de una solución de agrupación obtenida:

$$C_\rho = W_1 C_P^W + W_2 C_P^L \text{ tal que } W_1 + W_2 = 1$$

W_1 y W_2 denotan la importancia relativa del factor de carga computacional y de las comunicaciones interservidor comentadas anteriormente. En el caso general se tiene que $W_1=W_2 = 0.5$. Es evidente que, cuando el sistema DVE este funcionando

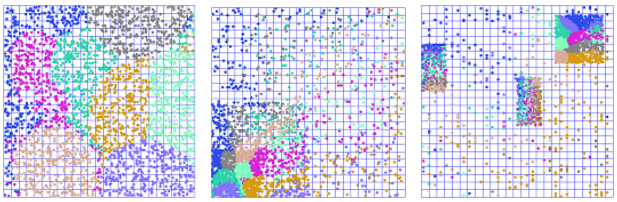
sobre una red de altas prestaciones (ej:Myrinet) el cociente W_1/W_2 sea mucho mayor que uno. Por el contrario, en redes lentas de medio compartido, o sobre Internet, este factor estará cercano a cero.
Estos mismos autores demuestran en [20] la NP-completitud del problema además de ofrecer a los investigadores una plataforma común de pruebas para evaluar sus soluciones y contrastar resultados. Además de esto, se describe un refinamiento de su algoritmo inicial, cuyas bases son publicadas en [19], así como una paralelización sencilla basada en criterios de volumen de avatares.
Existen otras aproximaciones al problema del particionado empleando diferente terminología ([18] y [19]). En [18] se detalla una solución que agrupa avatares en distribuciones topológicamente regulares. Para asegurar buenas prestaciones gráficas el número de distribuciones creadas se iguala al de servidores con lo que la correlación es directa. Esta solución no obtiene buenos resultados cuando los avatares no se encuentran distribuidos de manera uniforme.
Otra aproximación publicada ([19]) no basa el particionado en conceptos dinámicos como AOI, aura o locales. Para ello divide la escena 3D virtual en una malla regular. Sobre cada celda crea un grupo multicast ahorrando muchos mensajes. A pesar de ser una solución rápida y determinista, carga en exceso el rendimiento cuando los avatares se encuentran agrupados en zonas concéntricas.

IV. APROXIMACIONES EVOLUTIVAS

Todas las soluciones comentadas anteriormente para el problema del particionado, clasificado como NP-completo, comparten la misma característica. Son soluciones a medida definidas como heurísticas *ad-hoc* para resolver el problema. Debido a que ya en [20] se publican un conjunto de técnicas básicas no exhaustivas (basadas en procedimientos de inteligencia artificial) para abordar problemas de esta magnitud, se consideró interesante aplicar esta base de conocimiento en lugar de construir una solución desde el comienzo.
Dentro de estas técnicas, denominadas meta-heurísticas modernas, se encuentran los algoritmos evolutivos. Descritos en [23], son algoritmos de búsqueda no exhaustivos que resuelven problemas basándose en la evolución de procesos en la naturaleza. Los algoritmos descritos en este trabajo, fundamentados en técnicas de programación genética y enfriamiento simulado, son de este tipo. Dado que la eficiencia de una clasificación de avatares, para un DVE dado, es medida en términos de C_p, encontrar un mínimo en su espacio de soluciones equivale a incrementar las prestaciones del sistema a nivel de interactividad. Es aquí donde van a intervenir estos algoritmos, los cuales una vez adaptados, van a ofrecer por si solos soluciones de agrupamiento de clientes (avatares) en servidores en el mundo virtual 3D.

A. Elección de las configuraciones iniciales

La mayoría de metaheurísticas modernas basan su modo de funcionamiento en la generación de una o varias configuraciones (o presoluciones) iniciales. Durante la ejecución del procedimiento heurístico éstas van siendo transformadas en la solución final a través de una sucesión controlada de iteraciones. Tal y como se publica en los artículos originales de las metodologías tratadas en el presente trabajo ([22] y [24]), el conjunto de configuraciones iniciales de partida, para la resolución del problema, ha de ser generado de forma aleatoria. Esto conlleva, para el problema del particionado en DVE's, que en el instante inicial de comienzo del algoritmo, todo avatar va a estar preasignado a un servidor de forma totalmente aleatoria. Con ello, su valor hacia soluciones más óptimas, que reduzcan el coste del particionado global, será tanto más eficaz según sea la efectividad del método de clasificación. En la realidad, a pesar de estas consideraciones, la efectividad de una ejecución para la resolución de un problema en un conjunto de iteraciones dado, va a ser mucho mayor si el conjunto inicial de soluciones ha sido cuidadosamente elegido. La justificación es evidente. Estas metodologías, que buscan buenas particiones del sistema, basan su funcionamiento en una derivación intensiva de un conjunto de soluciones parciales. En toda la secuencia de iteración se busca que el coste de la última solución precalculada sea reducido al máximo evitando la caída en mínimos locales. Por ello, cuanto más evolucionada se encuentre esta configuración inicial, menor será el coste de particionado de las soluciones que pueda obtener. Basándose en la teoría de grafos para la clasificación de patrones ([21]), se ha diseñado un algoritmo inicial muy rápido que particiona de forma equilibrada los avatares de una escena virtual. La figura 2 muestra los resultados de este algoritmo inicial en la preclasificación de un DVE formado por 2500 elementos a repartir entre 8 servidores. Estos se localizan en el mundo virtual siguiendo tres diferentes distribuciones de probabilidad.



2a)Distrb.**Uniforme** $T_{ejecución}$= 1.32s 2b)Distrb.**Skewed** $T_{ejecución}$= 1.26s 2c) Distrb.**Clustered** $T_{ejecución}$= 1.28s

Fig. 2. Proceso de pre-asignación

B. Enfriamiento Simulado (SA) y Mapping

Método originalmente publicado en [22] que adopta conceptos físicos de la termodinámica. Simulated Annealing (SA) es un procedimiento basado en búsqueda local donde todo movimiento de mejora es aceptado y se permiten movimientos de empeoramiento de acuerdo con unas probabilidades. Aunque existen referencias en la literatura de la materia concernientes a diferentes problemas resueltos mediante SA ([23]), ha sido necesaria una adaptación y configuración previa para abordar, mediante esta técnica, el problema del particionado. Esta se basa en la evolución de una pre-solución inicial como la descrita anteriormente. Para ello, de forma iterativa, SA, durante un número de pasos dados (intentos), va a elegir 2 avatares de forma aleatoria entre los participantes en el DVE con la característica de que pertenezcan a servidores distintos. Una vez elegidos se intercambiarán entre ellos los servidores a los que pertenecen, computando tras esta permutación el coste C_p de la nueva partición obtenida. Si esta nueva solución reduce el coste de la anterior configuración, o a pesar de ser mayor lo es tan solo por debajo de un cierto umbral, esta permutación se mantiene. En caso contrario, como indica el algoritmo, se ha de deshacer el cambio y continuar con otra iteración.

A pesar de la aparente sencillez del proceso, son muchos los parámetros a configurar para la obtención de buenos resultados. Después de un análisis riguroso, se destacan 4 características básicas en la obtención de soluciones para el problema del particionado:

a) Avatares frontera. La elección de un avatar, de entre los que pertenecen a un mismo servidor no ha de realizarse aleatoriamente. Esta búsqueda no ha de ser libre, sino guiada, y ha realizarse sólo entre los *avatares frontera* de esa agrupación. Se entiende por avatar frontera a aquel que contiene en su AOI al menos un avatar alojado en un servidor diferentes al suyo. La justificación se basa en la forma en la que se realiza la búsqueda de la reducción iterativa de C_p. Debido a que un intercambio de avatares no modifica el factor equilibrado de cargas C_p^W, la reducción del coste total mediante permutaciones se consigue haciendo disminuir el coste de intercomunicación C_p^L. Por la forma en la que este sub-coste se calcula (evaluando las comunicaciones entre elementos cercanos), si un avatar, que estaba totalmente rodeado en un espacio virtual por vecinos asignados a su propio servidor, es asignado a otro servidor cualquiera, entonces, los costes de comunicación interservidor, transcurrido este cambio, se incrementarán.

b) Elementos intercambiables. A pesar de lo especificado por el método general, cabe la posibilidad de intercambiar más de dos elementos a la hora realizar cada iteración del método en la búsqueda del descenso de coste. La tabla 1 confirma que esto no debe realizarse:

Tabla 1. Análisis del número de elementos intercambiables

	Intercambio 2a2		Intercambio 3a3		Intercambio 4a4	
ENF.SIM.	**Cp**	**T(s)**	**Cp**	**T(s)**	**Cp**	**T(s)**
Uniforme	1707,62	6,35	1808,38	6,51	1817,90	7,02
Skewed	2628,46	13,79	2826,44	14,32	2992,30	15,82
Clustered	4697,61	29,62	5046,27	30,25	5283,70	33,98

c) Número de iteraciones. Factor clave en la aplicación eficiente de la técnica. Lógicamente, el aumento del número de iteraciones va a ser directamente proporcional al tiempo requerido para la ejecución e inversamente al coste final de la solución obtenida. El problema radica en que el factor de ambos procesos lineales es bastante diferente, por lo que se ha detectar un valor que a pesar de explorar exhaustivamente el árbol de decisiones, consiguiendo una solución de bajo coste, no requiera demasiado tiempo para ello. A su vez, debido a exigencias del tipo de pruebas a someter a las soluciones en el problema del particionado en DVEs ([13]), se ha de intentar que el valor para el parámetro elegido favorezca de forma ecuánime a las tres diferentes distribuciones sin perjuicio, en particular, para ninguna de ellas.

d) *Coeficiente de enfriamiento*. Determina la tolerancia del algoritmo a la aceptación de cambios desfavorables. Su variación no va repercutir en el incremento de los tiempos de ejecución del algoritmo ya que su descenso gradual, con el número de iteraciones, no interviene sobre manera en el coste temporal de la

obtención de soluciones. Por lo tanto, su elección se va a basar en la búsqueda de valores que planteen un comportamiento eficiente del algoritmo dando puntos de inflexión de costes para todas las distribuciones.
Debido a que la elección de estos parámetros, en concreto los dos últimos, va muy ligado a la batería de pruebas, se mostrará en la sección IV una configuración calibrada de estos.

C. Algoritmos Genéticos (AG)

Son algoritmos heurísticos basados en el concepto darvinista de evolución. Es muy abundante la literatura en la materia y sus principales aportaciones se basan en la solución de problemas complejos de diversa naturaleza. Para ello, crean poblaciones ([24]) de pre-soluciones que dejan evolucionar, por generaciones, hasta que obtienen post-soluciones de calidad aceptable. En publicaciones más recientes ([25]) se describen implementaciones y comparativas de la optimización de estos algoritmos mediante la paralelización de sus tareas.
Abordando el problema del particionado, la implementación propuesta en este trabajo se basa en partir de una población compuesta por un conjunto de elementos (cromosomas), cuya cardinalidad va a ser igual al número de soluciones parciales que simultáneamente va a manejar el AG en cada iteración. Cada uno de estos individuos vendrá representado por un vector de genes indicador, en un instante del tiempo, de una de las posibles configuraciones en la que el DVE a simular va a poder particionarse. Por lo tanto, cada uno de los elementos de este vector (descriptor de la asignación de un avatar para una solución) tomará un rango de valores igual al número de servidores que componen la simulación del DVE.
Diferentes factores influyen en la solución al problema del particionado con esta técnica:

a) *Soluciones iniciales.* Para evitar que el algoritmo pierda parte de sus iteraciones iniciales, probando combinaciones insatisfactorias, éstas han de ser lo más elaboradas posibles de forma que los costes C_p, de salida sean ya bajos. Para ello, se seguirán los mecanismo descritos en III.a.

b) Mecanismo de herencia basado en derivación. Obtenida la configuración inicial de la población y, tal y como se ha comentado anteriormente, AG va a fundamentarse en la realización, para cada iteración, de un conjunto de cruces dados entre los miembros de la población. Después de probar diferentes tipos de cruces entre ciudadanos se ha optado por utilizar un esquema de derivación o herencia para representarlos. Por lo tanto, cada iteración va a constar en la creación de un descendiente por cada uno de los N elementos de la población. De entre los 2N elementos obtenidos se elegirán, para pasar a la siguiente iteración, a los N que presenten menor coste C_p. Para crear una solución hijo a partir de otra padre se elegirán de ésta aleatoriamente dos elementos frontera con asignación de servidores distinta, las cuales serán intercambiadas. Este tipo de avatares, tal y como se comentó anteriormente, son los que con mayor probabilidad ofrecen permutaciones exitosas.

c) El proceso de mutación. Como la mayoría de heurísticas presentadas, AG ,en su exploración masiva del árbol de soluciones, corre el peligro de caer en mínimos locales, de los cuales por su mecanismo de trabajo sea difícil salir. Para evitar este posible fenómeno se ha incorporado también en la implementación de AG la mutación en el proceso del cruce de padres a hijos.
Una vez calculado el vector-hijo de una solución dada, se define la mutación como el proceso de sustituir aleatoriamente la asignación de servidor de uno de sus elementos por otro completamente distinto elegido al azar. Se han probado otro tipo de mutaciones, como realizar el cambio a más de un elemento, o cruzar las características de parejas de ciudadanos. Después de diferentes pruebas, la mutación basada en la alteración de un “gen” es la que ha ofrecido mejores soluciones. Cabe destacar que la producción de mutaciones es un fenómeno aleatorio controlado por un parámetro del AG, cuya variación en el sistema creará unas consecuencias en la obtención de soluciones.
d) Número de generaciones. De forma análoga a lo que ocurre en SA, en estos algoritmos se ha de fijar un valor de compromiso para el número de generaciones que permita disminuir, en la medida de lo posible, el coste de las soluciones obtenidas sin aumentar el exceso el tiempo de computación necesario para hallarlas.

V. RESULTADOS

Tal y como se comenta en la sección II, [13] recoge una importante contribución en el campo de los DVEs. En este artículo se ofrece a la comunidad investigadora en la materia una especificación común de creación de mundos virtuales. Mediante ella cualquier solución al problema del particionado pueda ser evaluada y comparada. Básicamente esta especificación define dos mundos virtuales básicos distintos. El primero de ellos, llamado SMALL, está formado

por 13 avatares en una escena virtual cuadrada (de 4x4 celdas lógicas) que han de ser asignados a 3 servidores diferentes. En segundo lugar, con el fin de evaluar el comportamiento de los algoritmos ante los diferentes tamaños de escena, se definen los mundos LARGE compuestos por 2500 avatares a clasificar en 9 servidores sobre zonas regulares de 25x25 celdas lógicas. Para ambos casos, se han de generar obligatoriamente tres tipos distintos de distribuciones de localización de avatares (uniforme, sesgada-skewed, y agrupada-clustered) con el fin de evaluar el comportamiento dinámico de las aplicaciones 3D sobre cada una de ellas.

Después de implementar ambos patrones de evaluación es necesario destacar 3 características inherentes de este test: a) se considera homogéneo y constante el ancho de banda entre todos los nodos de la arquitectura, ya sea entre tramos servidor-servidor o cliente-servidor; b) las características técnicas se consideran comunes a todos los servidores de las simulación; c) no es necesario ofrecer el tamaño de los escenarios virtuales en Km. o m ya que existe una relación que establece que si un avatar tiene un AOI de diámetro igual a D unidades, entonces las celdas regulares lógicas que forman la escena tendrán un área de D2 unidades.

Manteniendo estas especificaciones una batería de test, formada por cientos de diferentes DVEs, fue pasada a los diferente algoritmos en una aplicación prototipo. En ella, los avatares, siguiendo las diferentes distribuciones comentadas, fueron localizados aleatoriamente en la escena. A cada uno de estos avatares se le asignó una carga computacional modelada como un valor real entre 0 y 5. A su vez, para el cálculo de C_p, se cuantifico la relación de comunicaciones que une a dos avatares como un real (entre 0 y 5) inversamente proporcional a las distancia que les une (desde 5 en avatares muy cercanos y hasta casi 0 cuando las distancia que les separa esta ya cercana a D/2).

Cabe destacar que los resultados detallados a continuación, para ambos mundos virtuales, una vez fueron configurados ambas implementaciones evolutivas, se han realizado sobre equipos PC equipados con procesadores Pentium IV a 1.5Ghz, 256 MB de RAM y tarjeta gráfica nVidia Geforce2 MX- 400.

A. Resultados en mundos virtuales SMALL

Con el objetivo de detallar al máximo el tamaño del problema, y sólo para este tipo de mundos virtuales, fue posible reproducir una búsqueda exhaustiva que permitiera una visión global en la comparativa de métodos. Debido al reducido tamaño del modelo para estos mundos, una búsqueda completa requiere la exploración de 3^{13} (1.594.323) diferentes soluciones. Tal y como muestra la tabla número 2, es necesario gastar más de 4 segundos, en algunos casos, para asegurar que el algoritmo exhaustivo alcanza la solución óptima. A pesar de que estos tiempos de ejecución están lejos de las especificaciones temporales en tiempo real, la implementación de búsquedas exhaustivas es útil, en la medida que sea factible, como medida de evaluación a la eficiencia de las heurística implementadas para el problema.

	Computación t(s)	**Coste Sistema Cp**
Exhaustivo Alg.	3.411	6.54
Lui-Chan Alg.	0.0009	6.56
Enfriamiento Alg.	0.002	6.82
Genéticos Alg.	0.002	6.54

(a) Avatares en Distribución Uniforme

	Computación t(s)	**Coste Sistema Cp**
Exhaustivo Alg.	3.843	7.04
Lui-Chan Alg.	0.0010	8.41
Enfriamiento Alg.	0.005	7.46
Genéticos Alg.	0.003	7.04

(b) Avatares en Distribución Skewed

	Computación t(s)	**Coste Sistema Cp**
Exhaustivo Alg.	4.783	7.91
Lui-Chan Alg.	0.0011	10.56
Enfriamiento Alg.	0.006	7.91
Genéticos Alg.	0.005	7.91

(c) Avatares en Distribución Clustered

Tabla 2. Resultados en mundos virtuales **SMALL**

Al comparar ambas soluciones evolutivas con el algoritmo Lui-Chan, implementado para el mismo sistema según se detalla en [13], se puede observar claramente que se obtienen unas soluciones de mayor calidad a nivel de costes C_p del sistema .Del total de configuraciones evaluadas, tan solo Lui-Chan consigue rebajar las 6.82 unidades de coste con sus 6.56. A pesar de ello, ambas son superadas en eficiencia por la solución genética que consigue soluciones de coste 6.54 para esos patrones de prueba. A nivel de tiempo en la obtención de soluciones, ninguna solución evolutiva consigue, para mundos virtuales pequeños, ejecuciones más rápidas que las realizadas por el algoritmo Lui-Chan. Esto no plantea ningún problema ya que es necesario recordar que se está trabajando en un sistema 3D en tiempo real, donde supuesto un frame-rate exigido de 25fps se han de completar dibujos de escena a razón de 40ms. De estos 40ms, tras diferentes pruebas, una máquina con las características hardware descritas anteriormente necesita del orden de 12-15ms para realizar las tareas de procesamiento de polígonos y procesado

de píxel con el fin de dibujar una escena en un frame, sobrando 25-28ms a la etapa de aplicación en al que el algoritmo podría, entre otras cosas, ejecutarse si se toma de acción de ejecutar una vez por "frame" el algoritmo, decisión claramente pesimista. Estos últimos datos dependen en gran medida del tamaño de la escena en número de polígonos, tipo de representación a utilizar, texturizados, efectos y luces. Este hecho ya se contempla en [13] donde se afirma que para mundos virtuales SMALL soluciones por debajo de los 10ms son clasificadas como excelentes.

B. Resultados en mundos virtuales LARGE

En el caso de grandes DVEs (LARGE) la evaluación de los resultados obtenidos ha sido un poco diferente. En primer lugar, no existe un valor óptimo de coste Cp con el que comparar las diferentes metaheurísticas. El motivo de esta inexistencia se basa en el hecho de la imposibilidad de evaluar las 8^{2500} diferentes soluciones de una escena virtual de ese tipo. Además por [12], para esta configuración no se va a exigir que para cada uno de los "frames" se ejecute el algoritmo de partición, hecho que como se comprobará sería inviable.

La siguiente tabla muestra los resultados:

	Computación t(s)	**Coste Sistema Cp**
Lui-Chan Alg.	30.93	1637.04
Enfriamiento Alg.	6.35	1707.62
Genéticos Alg.	6.60	1832.21

(a) Avatares en Distribución Uniforme

	Computación t(s)	**Coste Sistema Cp**
Lui-Chan Alg.	32.17	3460.52
Enfriamiento Alg.	13.79	2628.46
Genéticos Alg.	14.59	2825.64

(b) Avatares en Distribución Skewed

	Computación t(s)	**Coste Sistema Cp**
Lui-Chan Alg.	43.31	5903.80
Enfriamiento Alg.	29.62	4697.61
Genéticos Alg.	29.20	4905.93

(c) Avatares en Distribución Clustered

Tabla 3. Resultados en mundos virtuales **LARGE**

Tal y como muestra la tabla 3, excepto para distribuciones uniforme de avatares, las soluciones evolutivas presentadas en este trabajo han logrado, en las diferentes pruebas a que se ha visto sometidas, soluciones con menor coste que las obtenidas por la heurística Lui-Chan. Para este caso de excepción, la solución basada en SA obtiene soluciones con un exceso de simplemente un 4.31% con respecto a las alcanzadas con la implementación Lui-Chan. Además de esta evidente mejora en cuanto a calidad de las soluciones obtenidas, las aproximaciones evolutivas son más rápidas, en el cálculo de agrupaciones, que las análogas Lui-Chan. Así consiguen reducir en un 79.5%, 57.13% y 32.58% los tiempos de ejecución para las distribuciones uniforme, sesgada (skewed) y focalizada (clustered), respectivamente.

Ante esto, se han de destacar dos importantes hechos. El primero de ellos se basa en que en grandes mundos virtuales, al contrario con lo que ocurría en mundos SMALL, el tiempo de ejecución en la aplicación del procedimiento de asignación de clientes-servidores es el parámetro más importante para mantener una eficiencia media en el sistema, ya que se ha de evitar mantener a un servidor (usualmente uno de los de la simulación) permanentemente calculando particiones. Por lo cual y debido a que todas las posibles soluciones son aproximaciones a la óptima, no se busca la mejor de todas sino aquellas que siendo suficientemente buenas se ejecuten más rápidamente.

Por otra parte, el hecho de que la distribución de avatares en un DVE convencional, por el comportamiento de los avatares en la escena, se realice usualmente en patrones sesgados (skewed) o focalizados (clustered) inclina, si cabe, todavía más la balanza hacía la eficacia de las soluciones presentadas en el este trabajo.

VI. CONCLUSIONES Y TRABAJO FUTURO

Los entornos virtuales distribuidos (DVE) se están convirtiendo en una importante área de interés dentro del campo del procesamiento paralelo y distribuido. Estos sistemas se caracterizan por una heterogeneidad inherente que aparece en numerosos puntos de su arquitectura. El control de sus clientes (avatares) interactuando en un mundo 3D compartido no es una tarea trivial.

En este trabajo, además de describir completamente la arquitectura en la que se diseñan estos sistemas se han presentado nuevas soluciones al problema de la asignación eficiente de clientes a los servidores de la simulación.

En lugar de desarrollar una técnica ad-hoc para este problema NP-completo, tal y como han realizado otros grupos de investigación, se han adaptado a las especificaciones de éste dos técnicas basadas en computación evolutiva.

Los experimentos realizados, sobre una plataforma de evaluación de mundos virtuales, demuestran que las técnicas presentadas obtienen soluciones de mejor calidad y en menor tiempo que las

implementadas siguiendo el paradigma Lui-Chan, máxima referencia en la literatura de la materia.

Con ello, las técnicas presentadas logran que los grandes mundos virtuales que las incorporen, como mecanismo base de particionado, presenten mejor escalabilidad.

Como trabajo futuro, el grupo investigador tiene planificada una doble vía de desarrollo. En primer lugar, adaptar otras metodología evolutivas, como colonia de hormigas, al problema del particionado en DVE. En segundo, siguiendo líneas de trabajo de grupos destacados en el campo de la computación evolutiva, se está trabajando en una implementación paralela a la solución genética. En su aproximación inicial aprovechará los servidores de la simulación al basarse en un esquema maestro-esclavo con comunicaciones asíncronas. La plataforma completa a desarrollar será portada para la implementación futura de un multi-simulador de conducción distribuido en el que el grupo investigador se encuentra actualmente trabajando.

VII. REFERENCIAS

[1] M.Abrash "Quake's game engine: The big picture"Dr. Dobb's. Journal.Spring. Year 1997.

[2] E. Alba, J.M. Troya "Analyzing synchronous and asynchronous parallel distributed genetic algorithms". Future Generation Computer Systems 17 (2001) 451-456.

[3] D.B.Anderson, J.W.Barrus, J.H.Howard, "Building multi-user interactive multimedia environments at MERL". IEEE Multimedia, 2(4):77-82, 1995.

[4] J.Barrus, R.Waters, D.Anderson "Locales: Supporting large multiuser virtual environments" IEEE Computer Graphics and Applications, vol 16.1996

[5] Steve Benford, Chris Greenhalgh, Gail Reynard, Chris Brown and Boriana Koleva. "Understanding and constructing shared spaces with mixed-reality boundaries". ACM Transactions on Computer-Human Interaction, vol. 5. no 3. pp 185-223. 1998.

[6] P.A.Berstein, V.Hadzilacos and N.Goodman." Concurrency, Control and Recovery in Database Systems". Addison-Wesley. 1997.

[7] P. Curtis, D.A. Nichols, "MUDs Grow Up: Social Virtual Reality in the Real World,", ftp://ftp.parc.xerox.com/pub/MOO/papers/MUDsGrowUp.ps. 1995.

[8] R.Duda, P.Hart, D.Stork. "Pattern Classification". Ed.Wiley Intescience Publication. 2000. p:567-580.

[9] E. Frecon, M. Stenius. "DIVE: A scaleable network architecture for distributed virtual environments". Distributed Systems Eng., vol- 5. 1998.

[10] Tomas A.Funkhouser. "Network Topologies for Scalable Multi-User Virtual Environments". Technical Report Bell Laboratories. 1996.

[11] Thomas A. Funkhouser "RING: A Client-Server System for Multi-User Virtual Environments" Symposium on Interactive 3D Graphics." .1996.

[12] F.C. Greenhlagh "Awareness -based communiction management in MASSIVE systems" Distributed Systems Engineering, vol 5, 1998.

[13] J.C.Hu, I.Pyarali, D.C.Schmidt, "Measuring the Impact of Event Dispatching and Concurrency Models on Web Server Performance Over High-Speed Networks," IEEE GIC, November.1997.

[14] S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi, "Optimization by simulated annealing," Science, vol. 220, pp. 671 680, 1983.

[15] P.V. Laarhoven, E. Aarts. "Simulated annealing: Theory and applications". Reidel Pub., Dordrecht, Holland. 1987.

[16] R.Lea, Y.Honda, K.Matsuda "Virtual Society: Collaboration in 3D space in Internet" Computer Supported Cooperative Working. Vol 6. 1997

[17] Michael Lewis and Jeffrey Jacboson " Game Engines in Scientific Research ". Communications of the ACM., Vol 45. Nº1 January 2002.

[18] Jhon C.S. Lui, M.F.Chan, Oldfield K.Y "Dynamic Partitioning for a Distributed Virtual Environment" Department of Computer Science & Engineering. The Chinese University of Hong Kong. Year 1998.

[19] John C.S. Lui, W.K. Lam, "General Methodology in Analysing the Performance of Parallel/Distributed Simulation under General Computational Graphs". ICSM 1999.

[20] Jonh C.S. Lui, M.F. Chan. "An Efficient Partitioning Algorithm for Distributed Virtual Environment Systems". IEEE TPDS,13-3, 03/2002.

[21] Michael R. Macedonia. "A Taxonomy for Networked Virtual Environments". IEEE Multimedia, 4(1) 48-56. January-March 1997.

[22] J.Mandeville, T.Furness, M. Kawahata, D.Campbell. "GreenSpace: Creating A distributed virtual environment for global applications". Proceedings of Networked Reality Workshop. 1995

[23] J.MaxField, T.Fernando and P.Dew "A distributed Virtual Environment for collaborative engineering". Presence Vol 7. 1998

[24] Z. Michalewicz, "Genetic Algorithms + Data Structures = Evolution Programs", Springer-Verlag, Second Edition, 1992.

[25] D.C.Miller, J.A. Thorpe. "SIMNET: The advent of simulator networking ". Proceedings of the IEEE, 83(8):1114-1123. Agosto 1995

[26] David L. Neyland, Virtual Combat: "A Guide to a Distributed Interactive Simulations", Stackpole Books, Merchanicsburg, PA. 1997

[27] T.Nitta, K. Fujita, S.Cono "An Application Of Distributed Virtual Environment To Foreign Language". Kansas . IEEE October 18 - 21, 2000

[28] J.M Salles, R.Galli, A.C.Almeida, C.A.C. Belo, J. Rebordão "mWorld: A Multiuser 3D Virtual Environment" IEEE CGraphics 97 (Vol. 17, No. 2)

[29] S.Singhal, M.Zyda. "Networked Virtual Environments" ACM Press, New York, 1999.

[30] R.Stuart. “The design of Virtual Environments”. Computing McGraw-Hill. Nueva York. 1996
[31] Peter T.S. Tam “Communication cost optimiz. and analysis in Distributed Virtual Environment”, Technical report RM1026-TR98-0412. 1998.
[32] Tam TszFirst. ”Term Paper New Architect. for DVE Object Based Dynamic Group Multicast Framework”. November 1997.
[33] Didier Verna, Yoann Fabre. “Urbi et Orbi: “Unusual Design and Implementation Choices for Distributed Virtual Environments” EPITA/LRDE 14-16. Kremlin. 2000.
[34] S.H.Zanakis, J.R.Evans, “Heuristic Optimization: Why, when and how to use it”. Interfaces, Vol. 11, nº 5, October. 1981.