IMPLEMENTACIÓN DE UNA TOPOLOGÍA HÍBRIDA PARA EL MANEJO DE COLISIONES EN UN AMBIENTE VIRTUAL

Diego Alejandro Álvarez Restrepo

Estudiante de Ingeniería de Sistemas, Integrante del grupo de investigación en Realidad Virtual.

Universidad EAFIT

Medellín, Colombia

dalvare6@eafit.edu.co

Helmuth Trefftz Gómez

Doctor en Ingeniería de la Computación – Universidad de Rutgers

Docente del departamento de Informática y Sistemas.

Director del laboratorio de Realidad Virtual

Universidad EAFIT

Medellín, Colombia

htrefftz@eafit.edu.co

RESUMEN

En este artículo se propone la creación de ambientes virtuales en red a partir de topologías híbridas, donde la mezcla entre cliente servidor y grupos multicast, permite disminuir los cuellos de botella y balancear las cargas a la hora de transmitir y controlar la información en la red. Uno de los factores más importantes en la creación de un ambiente virtual en red, es la topología de red a utilizar. Entre las topologías de red utilizadas con mayor frecuencia, se encuentran: la cliente servidor y la "par-a-par" (del inglés "peer-to-peer"), generalmente utilizando grupos multicast. Ambas topologías tienen sus ventajas y desventajas. Así, mientras el cliente servidor ofrece un muy buen grado de consistencia, la utilización de grupos multicast sacrifica dicha consistencia a favor de la velocidad, y reduce el número de mensajes que viajan por la red. El uso de una topología híbrida, como la propuesta en este artículo, puede ayudar a balancear las desventajas que presentan los esquemas puros cuando son implementados de manera individual.

ABSTRACT

This article reports the use of hybrid topologies (client-server and multicast groups together) in networked virtual reality environments to reduce the disadvantages presented individually by both of them. One of the more important issues to consider when building a networked virtual reality environment (NVRE) is the topology to use. The most commonly utilized topologies in NVRE's are client-server and peer-to-peer, generally built on multicast groups. The client-server

topology allows for a high level of consistency while multicast groups are faster and transmit less messages through the network, but is difficult to ensure the consistency. Hybrid topologies in virtual environments are useful to reduce the disadvantages presented by multicast and client-server topologies when implemented individually.

PALABRAS CLAVE

Ambientes virtuales, topologías híbridas, cliente-servidor, grupos multicast, realidad virtual en red.

KEY WORDS

Virtual environments, hybrid topologies, client-server, multicast groups, networked virtual reality.

INTRODUCCIÓN

Aunque factores tales como los tiempos de latencia y el ancho de banda encontrados en los ambientes virtuales en red, son difíciles de controlar, se espera que puedan ser mejorados con el advenimiento de Internet 2 (Beck, Micah and Moore, Terry 1998) y otras tecnologías, que prometen brindar muy buenas prestaciones a los ambientes virtuales. Sin embargo, mientras que esta nueva red se vuelve de uso común, e incluso en un futuro próximo cuando ya lo sea, será muy probable que problemas como los cuellos de botella representados por los

servidores ante el arribo de una mayor cantidad de mensajes, sean aún caso de estudio de muchas investigaciones, e incluso puedan adquirir mayor relevancia ante los factores mejorados por tecnologías futuras. Por tal motivo, es importante enfocarse en factores tales como la topología de la red y los tipos de datos a transmitir a través de ella.

Cuando se piensa en una topología en red para un ambiente virtual, lo más común es elegir, de acuerdo con la aplicación, la utilización de grupos “multicast”, “broadcast”, esquemas punto a punto o modelos cliente servidor, y evaluar sus ventajas y desventajas.

Una topología cliente servidor, como la propuesta por Funkhouser en su sistema RING (Funkhouse, 1995), ofrece un alto nivel de consistencia en el mundo, pero sacrifica un poco la velocidad. Además, si la cantidad de mensajes generados en el sistema es muy alto, el servidor tomará mayor tiempo en procesarlos y replicarlos, convirtiéndose en un gran cuello de botella para la aplicación.

Por otra parte, en sistemas que recurren al uso de grupos multicast, aumenta el número total de mensajes que se pueden procesar, pero se hace más difícil controlar factores como la consistencia, la transmisión segura de datos y el manejo de colisiones.

Entonces es importante buscar un equilibrio entre unas y otras topologías, con el fin de balancear las cargas en el transporte de los datos, conllevando a un aumento en los beneficios generales del sistema, tales como una mejor detección

de colisiones y una rápida actualización de mensajes de posición de los usuarios, y a una disminución de las desventajas que las topologías cliente-servidor y multicast, presentan cuando actúan de manera individual.

TRABAJOS RELACIONADOS

Uno de los factores que se analizan cuando se desea crear un ambiente virtual en red, es la topología de red a utilizar. Topologías cliente servidor como las utilizadas por aplicaciones tales como RING (Funkhouser,1996) y NetEffect (Diaper, 2001), son utilizadas ampliamente y de gran aceptación. Sin embargo, la capacidad del servidor impone límites a la aplicación.

Por otra parte, en topologías punto a punto como las que utilizan DIVE (Distributed Interactive Virtual Environment, Carlson, 1993) y NPSNet (Macedonia, 1995), el ancho de banda y el tiempo de latencia en la red introducen los límites a la aplicación.

Las topologías enunciadas, fueron mezcladas e introducidas en la creación de aplicaciones Cliente–Servidor y punto a punto, tales como DIVE con DiveBone (Diaper, 2001) y SPLINE (Barrus, 1996) e incluso en aplicaciones comerciales como la MNC (Multinational Corporation, Savolainen, 1997), que utiliza una topología Cliente-Servidor con sus usuarios y una punto a punto entre sus servidores.

Además, la aplicación MASSIVE-1 (Model, Arquitecture and System for Spatial Interaction in Virtual Environments, Greenhalgh, 1995) que utiliza una topología punto a punto unicast, hace uso de un llamado “servidor de auras”, que permite a

los usuarios interactuar solamente cuando están en el mismo mundo y muy cerca dentro del espacio virtual.

Recientemente, las arquitecturas híbridas se han utilizado también para resolver el problema de la heterogeneidad en el poder de cómputo de los computadores que participan en sesiones de Realidad Virtual Distribuida. Un ejemplo notable es la arquitectura VELVET (Oliveira, Jauvane C. de, Georganas, Nicolas D (2003)).

Mientras este artículo se encontraba en evaluación apareció un trabajo similar, en el cual se propone utilizar múltiples grupos “multicast” para lograr una mayor escalabilidad en el número de participantes simultáneos en un ambiente virtual distribuido.

En este artículo se propone la creación de una topología híbrida que permita reducir los cuellos de botella en el servidor, mejorar a la vez la detección de colisiones y realizar una actualización rápida de la posición de los clientes en el mundo virtual, mediante el uso de un grupo multicast, un servidor y los clientes que se matriculan a ambos.

DEFINICIÓN E HIPÓTESIS

En este artículo se reportan los resultados de utilizar una topología híbrida conformada por un grupo “multicast”, en combinación con un modelo cliente servidor. En dicha topología el servidor actúa como árbitro o juez, que se encarga de decidir cuándo dos o más usuarios colisionan con objetos del mundo, y decidir cuál colisionó primero.

Los clientes, por su parte, no tienen que estar enviando las actualizaciones de posición a través del servidor, sino que utilizan el grupo “multicast” para comunicarse con los demás clientes. El concepto de fondo es simple: “El servidor no se entera de lo que no le interesa”.

Aunque el servidor hace parte del grupo “multicast”, se comunica directamente con los clientes sólo para manejar las colisiones. La topología busca balancear el tráfico de mensajes entre ambos esquemas, para evitar que el servidor se convierta en un cuello de botella. Por otra parte, se busca una mayor escalabilidad para el sistema.

Las hipótesis establecidas para llevar a cabo lo anterior serán:

1. *El número promedio de mensajes de reparación de colisiones generado en la topología híbrida, es menor que el número promedio de estos mensajes, generados en una topología pura, sea Cliente - Servidor o Multicast.*

Entiéndase por mensaje de reparación, todo aquel mensaje que se transmite a través de la red para informar a un cliente que su colisión con un determinado objeto del mundo no es válida, puesto que otro cliente colisionó con dicho objeto antes que él.

2. *En promedio, el número total de mensajes generados para una cantidad fija de usuarios en una topología híbrida, es menor que el promedio total de mensajes generados en una topología cliente-servidor.*

METODOLOGÍA

Se realizaron pruebas a tres aplicaciones diferentes en una red LAN de 10 Mbps. Las tres consisten en un mundo virtual donde los usuarios, que son agentes simples que simulan el comportamiento de un usuario real, son representados por cubos y deben llegar a ciertos puntos de control, o bases, representados por conos (ver Figura 1). El primero en llegar a un cono reporta su colisión a los demás miembros de la red, para indicar que él es el nuevo dueño de la base. Pero ¿qué pasa si dos o más usuarios llegan al mismo tiempo a una base? La colisión debe solucionarse en favor de uno de ellos. A este concepto lo llamamos *reparación*. Si dos o más clientes reclaman que han llegado a colisionar un mismo objeto, es necesario resolver de alguna manera el conflicto. En el caso de una arquitectura cliente servidor, por ejemplo, una aplicación que corra en el servidor puede detectar esta situación y asignar la colisión al objeto cuyo mensaje de

reclamación llegó primero. En este caso se le envía un mensaje a los nodos cuyos mensajes llegaron posteriormente, invalidando sus colisiones.

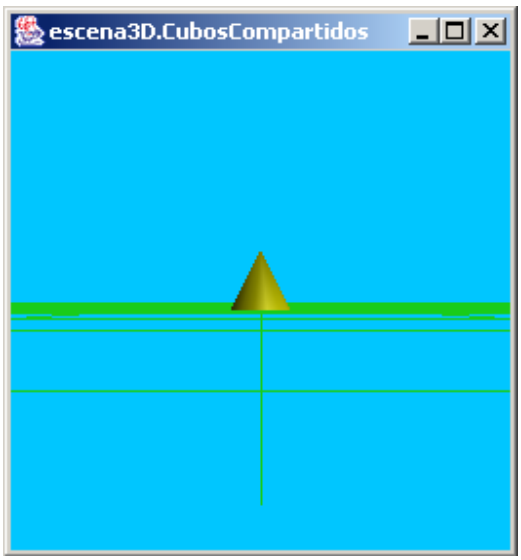

Figura 1. Ambiente virtual de prueba.

Para solucionar el problema de las colisiones, la aplicación fue diseñada bajo tres topologías diferentes:

La primera es una topología cliente – servidor, en la que el servidor decide quién se queda con la base. Además el servidor sirve de puente entre todos los clientes y a través de él pasan los mensajes de actualización de posición de los usuarios, cada vez que éstos se mueven. La comunicación es a través del protocolo TCP (Peterson, Larry and Davie, Bruce S., 2000).

La segunda es una topología basada en un grupo “multicast” donde no existe un agente central que decida la colisión, sino que los usuarios tienen que transmitir el

mensaje de colisión a todos los demás y esperar a que éstos le retransmitan diciendo que están de acuerdo en que él quede como dueño de la base. Si al menos uno de los otros usuarios no responde o responde negativamente a la petición, el usuario inicial deberá esperar un tiempo aleatorio y retransmitir su petición, tal como lo hace el algoritmo CSMA/CD (Tanenbaum, 1997). A través del grupo multicast pasan también los mensajes de actualización de posición de todos los clientes. La comunicación es a través del protocolo UDP (Peterson, Larry and Davie, Bruce S., 2000).

La tercera topología es un esquema híbrido formado por un grupo “multicast”, los clientes y un servidor. Aquí las colisiones son solucionadas por el servidor vía TCP, y los mensajes de actualización de la posición de los clientes pasan a través del grupo “multicast” vía UDP.

Se realizaron las pruebas para las tres topologías en las mismas máquinas, de la siguiente manera:

Cinco pruebas para 2 clientes en la topología cliente-servidor, luego otras cinco pruebas incrementando el número de clientes a 3, y así sucesivamente, incrementando el número de usuarios cada cinco pruebas hasta llegar a un máximo de 7.

Los resultados de las pruebas se describen en el Anexo.

La prueba se realizó de la misma manera para las otras dos topologías, obteniendo de cada una los datos correspondientes los que se analizan a continuación.

ANÁLISIS DE RESULTADOS

Los datos recabados fueron promediados de acuerdo con el número de clientes, así se obtuvo el promedio del número de reparaciones y del número total de mensajes de la prueba, según el número de clientes de la misma.

1. Análisis del número de reparaciones

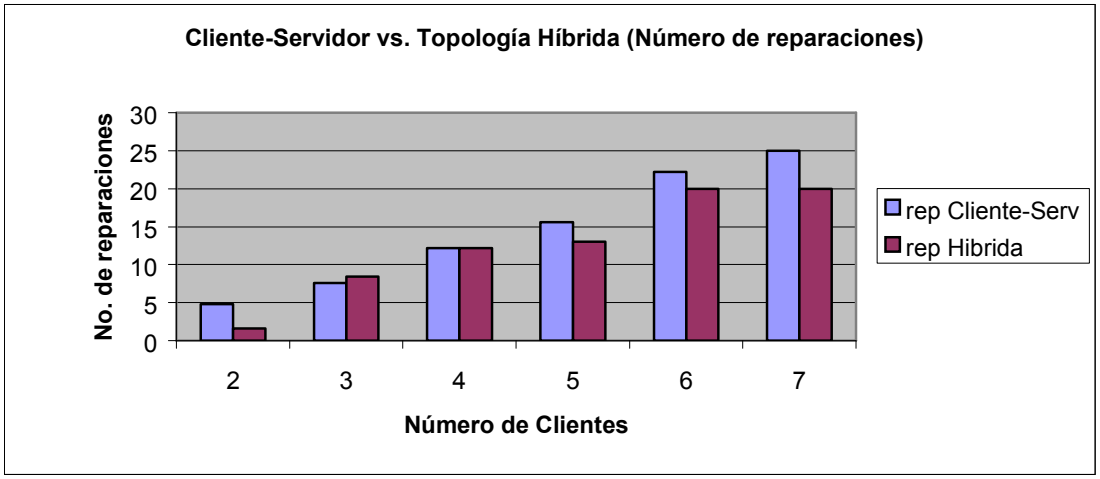

Figura 2. Número de reparaciones,

Cliente-Servidor vs. Topología Híbrida.

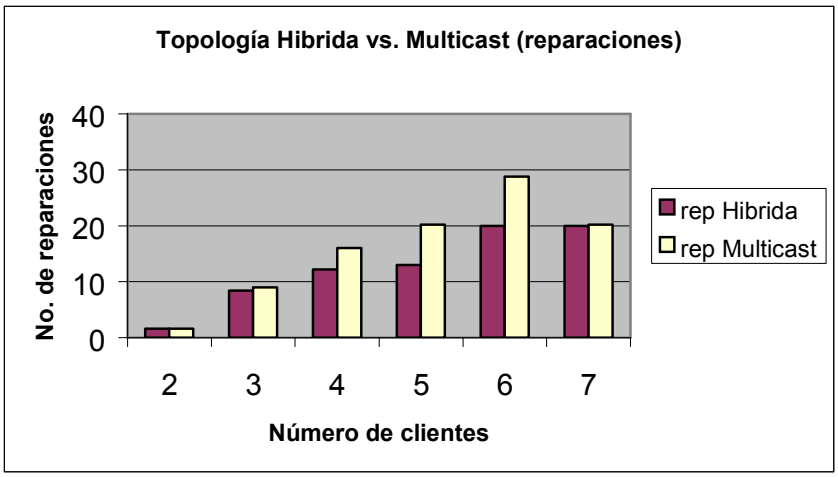

Figura 3. Número de reparaciones,

Topología Híbrida vs Multicast

Como se puede observar, el número de mensajes de reparación es menor en la topología híbrida en comparación con la topología "multicast" (figuras 1 y 2). Igualmente, el número de dichos mensajes es en general menor en la topología híbrida, al ser comparado con la topología cliente-servidor. Aquí, aunque ambas topologías utilizan el mismo servidor para solucionar las colisiones, la diferencia está dada por el número de mensajes que debe procesar dicho servidor, ya que, como se verá a continuación, el número de mensajes que pasan por el servidor en la topología Cliente-Servidor, es mucho mayor que el número de mensajes que debe procesar el servidor en la topología híbrida, lo cual permite que en esta última las colisiones se

solucionen más rápidamente, debido a la poca congestión de mensajes en el servidor.

2. Análisis del total de mensajes.

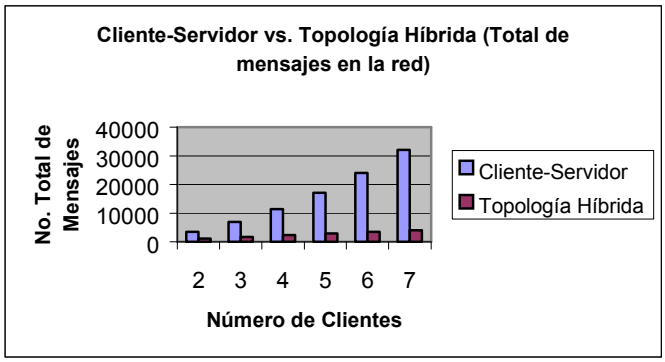

Figura 4. Total de mensajes en la red, Cliente-Servidor vs. Topología híbrida

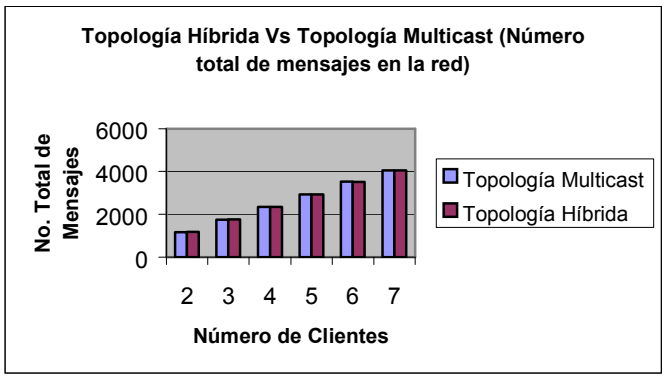

Figura 5. Total de mensajes en la red, Topología Multicast vs. Topología híbrida

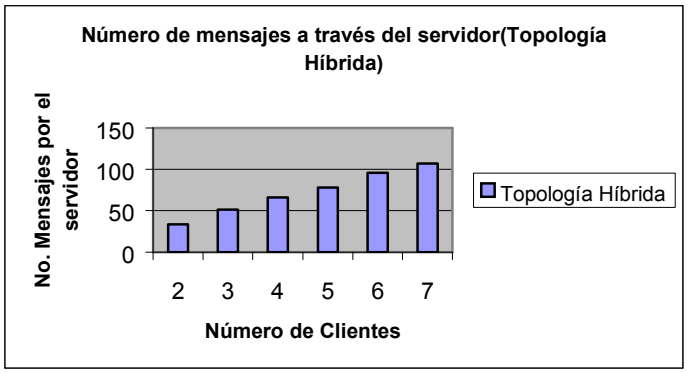

Figura 6. Total de mensajes a través del servidor en Cliente-Servidor

Como se puede observar en las figuras 4, 5 y 6, el número de mensajes totales que se generan y envían por la red durante el tiempo de ejecución de la prueba, es mucho menor en una topología híbrida, si se compara con una topología Cliente-Servidor, y es aproximadamente igual al número de mensajes totales que se generan en una topología multicast.

Además, en la figura 6 se observa que el número de mensajes que pasan a través del servidor en una topología híbrida, no llegó nunca a 150, mientras que el número de mensajes que pasan a través del mismo servidor en una topología Cliente-Servidor es del orden de hasta los 31.000 (dato tomado de la gráfica 4). Esta diferencia en el número de mensajes, permite que en una topología híbrida el servidor se encuentre menos congestionado y que pueda tomar las decisiones acerca del manejo de colisiones de forma más rápida.

CONCLUSIONES

Los resultados obtenidos durante las pruebas, muestran que con el uso de una topología híbrida se reduce el número promedio de mensajes de reparación que deben generarse, en comparación con una topología Multicast. Además, el número promedio de mensajes totales generados en una topología híbrida es mucho menor que el número total de mensajes que se generan en una topología Cliente-Servidor.

Durante la ejecución de las pruebas había máquinas muy rápidas que opacaban el desempeño de las demás, apoderándose rápidamente de las bases. Esto fue tratado por software mediante la adición de retardos para que el cliente que primero llegara a las proximidades de alguna base, se detuviera y esperara a que un segundo cliente se encontrara tan próximo como él a colisionar. Así el sistema se hizo más homogéneo.

El número de pruebas hechas por cada número de clientes fue baja (cinco), lo cual podría afectar el resultado general de las pruebas.

En las aplicaciones probadas se adicionaron tiempos de retardo de 100 milisegundos, con el fin de simular posibles congestiones en la red. La adición se realizaba entre el momento en que un cliente colisionaba con una base y el momento en el cual se empezaba a solucionar la colisión para dicha base. Esto permitió observar el número de colisiones generadas en el peor de los casos, lo

cual indica que los sistemas funcionarán de mejor manera en redes de poca congestión y que los resultados de la investigación son aplicables hasta cuando en la red haya un tiempo de latencia de 100 milisegundos entre los mensajes.

El tiempo de 100 milisegundos fue tomado a partir de “Networked Virtual Environments” (Singhal, 1999), donde se habla de 100 milisegundos como el tiempo mínimo de latencia para las conexiones a Internet por módem, a través del sistema telefónico, y donde se afirma además que este tiempo es muy amplio para permitir realizar comunicaciones interactivas en un ambiente virtual, dado que permitiría hacer muy pocas actualizaciones por segundo.

Los frutos de esta investigación pueden ser utilizados, entre otras, en áreas como la realidad virtual distribuida y la programación de juegos en red, donde se requiere un manejo rápido y consistente de las colisiones.

Es importante anotar, además, que una topología híbrida es fácil de implementar en comparación con una topología multicast pues, en esta última, si se desea construir una buena solución para el manejo de mensajes y colisiones, el analista debe tener buenos conocimientos en cuanto al manejo de sistemas distribuidos.

TRABAJO FUTURO

La utilización de topologías híbridas puede ser extendida a una mayor cantidad de mundos virtuales. Estos pueden ser partidos en celdas de manera estática o dinámica y asignados a diferentes servidores de colisiones, mientras que los

mensajes de actualización de posición de los usuarios pueden continuar siendo enviados a través de un grupo “multicast”. El grupo multicast será el encargado de replicar los mensajes de actualización de la posición de los usuarios a través de la aplicación.

Agradecimientos

Agradecemos al grupo editorial de la revista Eafit y a los evaluadores. Sus comentarios mejoraron notablemente la calidad de este artículo.

BIBLIOGRAFÍA

Barrus, John; Waters, W.; Anderson, R.C. and David B. (1996) “Locales and Beacons: Efficient and Precise Support For Large Multi-User Virtual Environments”. Proceedings of VRAIS'96, Santa Clara, CA.

Beck, Micah and Moore, Terry (1998). “The Internet2 Distributed Storage Infrastructure project: an architecture for Internet content channels”. In *Computer Networks and ISDN Systems*. Volume 30, Number 22-23. pp. 2141 – 2148.

Carlson, C. and Hagsand, O (1993) “DIVE: A Platform For Multi-User Virtual Environments”. In: *Computer and Graphics*, 17(6). pp. 663-669.

Cohen, J.; Lin, D.; Ming, C. ; Manocha, Dinesh and Ponamgi, Madhav K. (1995) “I-COLLIDE: An Interactive and Exact Collision Detection System for Large-Scale Environments”. In: *Symposium on Interactive 3D Graphics*. pp. 189–196.

Diaper, Dan and Colston, Sanger (2001) “Digital places and spaces for interaction”. In: Elizabeth F. Churchill, David N. Snowdon and Alan J. Munro (Eds). *Collaborative Virtual Environments*. 344 p.

Drew, Richard; Morris, David; Dew, Peter and Leigh, Christine (1996). “A system architecture for supporting event based interaction and information access”. University of Leeds. Leeds, England. http://citeseer.nj.nec.com/370116.html. Noviembre de 2002.

Funkhouser, Thomas A. (1995). “RING: A Client-Server System for Multi-User Virtual Environments”. In: *ACM SIGGRAPH Special Issue on 1992 Symposium on Interactive 3D Graphics*. Monterrey, CA, pp. 85-92.

Funkhouser, Thomas A. (1996) “Network Topologies for Scalable Multi-User Virtual Environments”. In *VRAIS 1996*. pp. 222 – 229.

Greenhalgh, Chris and Benford, Steve (1995). “MASSIVE: a Distributed Virtual Reality System Incorporating Spatial Trading”. In: *15th International Conference on Distributed Computing Systems* (DCS`95), May 30 June 2 1995. Vancouver, Canada. IEEE Computer Society Press.

Held, Martin; Klosowski, James; T., Mitchell and Joseph S.B. (1995). *Evaluation of Collision Detection Methods for Virtual Reality Fly-Throughs*. Department of Applied Mathematics and Statistics. Stony Brook: State University of New York, Stony Brook. pp 205-210.

Macedonia, Michael R.; Zyda, Michael J.;Pratt, David R.; Brutzman, Donald P. and Barham, Paul T (1995) “Exploiting Reality with Multicast Groups: A Network

Arquitecture for Large Scale Virtual Environments". In: *Proceedings of the 1995 IEEE Virtual Reality Annual Symposiu*m*,* North Carolina.

Oliveira, Jauvane C. De; Georganas, Nicolas D. (2003). "VELVET: an adaptive hybrid architecture for very large virtual environments". In: *Presence: Teleoperators and Virtual Environments*. Vol 12, Num 6. pp. 555 – 580. 2003.

Peterson, Larry and Davie, Bruce S. (2000). *Computer Networks*. 2nd Ed. Morgan Kaufmann. 748 p.

Savolainen V.; You Y.; Zhang Zh. (1997). *Virtual Teamwork Environment - A Web-Based Conical Administration Framework for MNC*. Department of Computer Science and Information Systems. University of Jyväskylä, PL 35, Finland.

Singhal, Sandeep and Zyda Michael (1999). *Networked Virtual Environments. Desing and Implementation*. Addison-Wesley. 331 p.

Tanenbaum, Andrew S. (1997). *Redes de Computadoras*. Pearson. 3 Ed.. 658 p.

Léty, Emmanuel; Turletti, Thierry and Baccelli, Francois (2004) "SCORE: a scalable communication protocol for large-scale virtual environments". In: *IEEE/ACM Transactions on Networking*. Vol 12, Num 2. April, 2004. pp. 247 – 260.

Topol, Brad (1998). "Robust State Sharing for Wide Area Distributed Applications". *International Conference on Distributed Computing Systems*. Atlanta, Georgia: College of Computing Georgia Institute of Technology.

Yang, J. and Lee, D. (2000). "Scalable Prediction Based Concurrency Control for Distributed Virtual Environments". In: *Proceedings of IEEE VR 2000*.

ANEXOS

RESULTADOS DE LAS PRUEBAS

Los siguientes son los resultados detallados de las pruebas realizadas, de acuerdo con el número de usuarios.

Se observa la cantidad de mensajes de reparación que se produjeron durante la prueba, el número de mensajes totales, y el tiempo de ejecución de la misma

1. Resultados topología Cliente-Servidor

		CLIENTE-SERV	
N° usuarios	Msg_reparación	MensajesTotal	Tiempo(mili)
2	5	3481	29893
2	4	3474	29212
2	6	3476	29202
2	5	3475	29212
2	4	3474	29202
3	10	6945	29452
3	7	6926	29413
3	10	6937	29382
3	6	6913	29212
3	5	6908	29223
4	12	11461	28972
4	11	11455	28972
4	13	11472	28982
4	13	11467	28992
4	12	11471	28962
5	24	17249	29031
5	8	17161	29062
5	12	17165	28951
5	22	17205	28962
5	12	17171	28981

6	22	24052	28992
6	24	24061	29002
6	22	24045	28961
6	22	24052	28962
6	21	24044	28962
7	23	32014	28972
7	24	32031	29002
7	23	32022	28982
7	27	31962	28991
7	28	32051	29042

2. Resultados de la topología híbrida

   Para esta prueba se tomaron además el número de mensajes que pasaron a través del servidor y a través del grupo multicast.

HÍBRIDA

N°_usuarios	Mgs_reparación	Msg_multicast	Msg_server	Total_Msg	Tiempo(mili)
2	2	1146	34	1182	29613
2	1	1147	33	1181	29663
2	1	1144	33	1178	29622
2	2	1144	34	1180	29593
2	2	1146	34	1182	29793
3	10	1701	53	1764	28771
3	10	1703	53	1766	28811
3	2	1700	45	1747	28791
3	10	1702	53	1765	28781
3	10	1700	53	1763	28792
4	10	2267	64	2341	28811
4	20	2275	74	2369	28811
4	11	2273	65	2349	28831
4	10	2268	64	2342	28771
4	10	2272	64	2346	28792
5	10	2827	75	2912	28791
5	10	2832	75	2917	28781
5	20	2842	85	2947	28812
5	15	2832	80	2927	28821
5	10	2829	75	2914	28811
6	20	3388	96	3504	28811

6	20	3394	96	3510	28821
6	20	3385	96	3501	28801
6	20	3390	96	3506	28811
6	20	3385	96	3501	28801
7	20	3919	107	4046	28782
7	20	3928	107	4055	28822
7	20	3920	107	4047	28801
7	20	3916	107	4043	28812
7	20	3917	107	4044	28822

3. Resultados de la topología multicast

MULTICAST			
N° usuarios	Msg_reparación	Msg_multicastTotal	Tiempo(mili)
2	2	1170	29773
2	3	1173	29793
2	1	1166	29762
2	0	1166	29753
2	2	1170	29783
3	5	1742	29001
3	10	1753	28951
3	10	1753	28932
3	10	1752	28932
3	10	1751	28942
4	34	2383	29002
4	10	2329	28942
4	16	2345	28982
4	10	2334	28982
4	10	2330	28951
5	29	2947	28951
5	0	2885	28882
5	10	2904	28912
5	28	2950	29001
5	34	2961	29032
6	12	3486	28891
6	23	3515	28942
6	23	3502	28941
6	54	3584	28981
6	32	3523	28962
7	20	4051	28971

7	20	4046	28902
7	27	4066	28942
7	23	4073	28952
7	11	4031	28941