Protocolo de comunicaciones para control de la generación distribuida de flujo multimedia

C. F. Perez-Monte[1,2], M. F. Piccoli[2], M. Perez[1], C. Luciano[3,4], S. Rizzi[5]

[1]GridTICs – Universidad Tecnológica Nacional - Facultad Reg. Mendoza (UTN-FRM)
Mendoza, Argentina

[2]LIDIC – Universidad Nacional de San Luis (UNSL)
San Luis, Argentina

[3]Dept. of Biomedical and Health Information Sciences
[4]Dept. of Bioengineering - University of Illinois at Chicago (UIC)
Chicago, USA

[5]Argonne Leadership Computing Facility – Argonne National Laboratory (ANL)
Chicago, USA

cristian.perez@gridtics.frm.utn.edu.ar, mpiccoli@unsl.edu.ar,
mauricio.david.perez@gmail.com, clucia1@uic.edu, srizzi@alcf.anl.gov

Resumen. *Aplicaciones de tiempo real como realidad virtual generan un flujo multimedia de tipo de máquina-humano que puede ser dividido tanto espacial como temporalmente para distribuir recursos de procesamiento, memoria o red. Este trabajo describe un protocolo para control de flujo de datos multimedia, donde su generación debe ser distribuida para incrementar el rendimiento. Este protocolo no describe la comunicación del flujo multimedia, sino solamente la comunicación de control para la generación de un único flujo multimedia entre múltiples nodos teniendo en cuenta un esquema de mejor esfuerzo. Para lograr estos objetivos se propone un protocolo de capa de aplicación, el cual utiliza UDP o UDP-lite en capa de transporte, IPv6 multicast en capa de red y Gigabit Ethernet con control de flujo en capa de enlace de datos.*

1. Introducción

Diversos protocolos se han propuesto para el transporte de información multimedia requerida para la comunicación humano-humano, ejemplos son el video o el audio. Existe un área específica de IETF denominada ART [IETF 2016] con grupos de trabajo dedicados al continuo desarrollo de los mismos. En la actualidad, aplicaciones como la realidad virtual, permiten el transporte de información multimedia para la comunicación humano-máquina, utilizado ampliamente en la industria del entretenimiento [Huang 2013]. En dichas aplicaciones, generalmente existe una comunicación de ancho de banda reducido, correspondiente a la comunicación humano a máquina y una comunicación de ancho de banda elevada, correspondiente al flujo multimedia de video de máquina a humano. Dicho flujo multimedia, para ser foto-realista, debe poseer técnicas de generación de la escena complejas y, además, poseer resolución y cantidad de cuadros por unidad de tiempo elevadas para una fluidez aceptable. Para cumplir con estos requisitos se necesita una elevada capacidad de procesamiento, una apropiada

solución es la utilización de numerosos nodos para la generación del flujo de video multimedia.

En el presente trabajo se propone un protocolo de comunicaciones para la sincronización de tareas en la generación de un flujo de video multimedia. El desarrollo de un protocolo de comunicación para controlar un flujo multimedia en tiempo real y en un sistema de múltiples nodos no es una tarea trivial, por lo cual se sugiere su discusión en grupos de IETF para su estandarización. En este trabajo se propone sentar las bases para iniciar su discusión, por ello el protocolo de comunicación fue simplificado, optimizado y generalizado para su utilización en un mayor número de aplicaciones. Representa una actualización de un trabajo anterior [Perez-Monte 2014] donde se mostró su viabilidad a través de resultados experimentales.

El presente trabajo se divide de la siguiente manera: en primer lugar, en esta sección desarrollamos conceptos básicos de la renderización distribuida y trabajos relacionados. En la sección 2, describimos un sistema de procesamiento paralelo donde los nodos tienen diferentes roles, y un modelo de distribución de tareas, los cuales son el punto de partida para el diseño del protocolo de comunicaciones. Finalmente, en la sección 3 exponemos el protocolo de comunicaciones.

1.1. Renderización en Tiempo Real y Foto-realista

Un modelo 3D es una representación de un objeto 3D, el cual puede ser descripto a través de polígonos o de un volumen compuesto de voxels u otra técnica. Renderización es un término frecuentemente utilizado para describir el proceso por el cual a un modelo 3D se le realiza un procesamiento por el cual, mediante determinados parámetros de entradas, se genera una imagen 2D desde un punto de vista del observador. Tanto la ubicación del observador como su orientación visual son los parámetros de entrada más importantes, sin embargo pueden existir otras como la ubicación de la iluminación. En la renderización en tiempo real, los parámetros de entrada pueden variar constantemente debido al cambio constante de posición del observador o las luces, implicando una generación de imágenes 2D en el tiempo. Además de los parámetros de entrada, el modelo 3D determinará un estado interno del método de renderización, el cual puede cambiar si la escena cambia o los objetos constitutivos sufren desplazamientos o deformaciones.

Así como la renderización tiene como finalidad generar una imagen para la visualización de un modelo 3D, la renderización foto-realista tiene como objetivo, además, generar una imagen con una realidad visual lo más cercana a la realidad usando técnicas visuales de iluminación y sombreado demandantes de un procesamiento elevado. La misma tiene gran aplicación en áreas como el entretenimiento, entrenamiento médico, planeamiento prequirúrgico entre otras.

Una mayor capacidad de procesamiento puede obtenerse a partir de sistemas paralelos constituidos por grupos de computadoras conectadas en red. Para poder aprovechar la capacidad de dichos sistemas se debe recurrir a técnicas de renderización paralelas, descriptas a continuación.

1.2. Renderización Paralela Distribuida

En renderización paralela distribuida, la taxonomía de Molnar [Molnar 1994] [Cox 1995] establece dos tipos de técnicas basadas en la manera en cómo distribuye el trabajo

entre los diferentes nodos de un sistema distribuido. Estas dos técnicas se conocen con el nombre de *Sort first* o *2D* y *Sort last*, sus principales características son:

- La renderización paralela distribuida *Sort first* o *2D* propone la división de la pantalla en áreas no solapadas, asignando una a cada nodo o unidad de cómputo disponible. De esta forma, cada nodo renderiza una sección determinada de la pantalla completa. Ha sido aplicado en numerosos sistemas [Schwarz 2010] [Moloney 2011].

- En la técnica *Sort last*, la renderización es hecha dividiendo sectores de volúmenes o primitivas, aplicando la renderización y realizando la composición de los resultados parciales para obtener la imagen final. También ha sido ampliamente aplicado. [Eilemann 2007][Marchesin 2008].

Una tercera técnica de renderización distribuida fuera de la taxonomía de Molnar es *Alternate Frame Rendering* (*AFR*), muy utilizada en ambientes con varias GPUs en una única PC [Diard 2009][Monfort 2009]. Se basa en la distribución de cuadros de pantalla.

De las tres técnicas citadas, *2D* y *AFR* pueden ser aplicadas con el sistema propuesto en este trabajo, constituyendo una buena solución para la aplicación en la renderización distribuida foto-realista en tiempo real. También es posible aplicar otras técnicas, propias de algunos algoritmos de renderización. Además, todas pueden combinarse y definir nuevas políticas aplicables a ambientes distribuidos-paralelos para resolver problemas con gran demanda computacional.

1.3. Sistemas de Renderización Causales

Un sistema de renderización es causal o determinístico cuando, ante parámetros iniciales de entrada y de estado interno, es perfectamente determinable la imagen de salida renderizada. El sistema de renderización propuesto en la próxima sección tiene estas características. Además podemos determinar dos tipos de sistemas de renderización causales, ellos son:

- Determinista externo: Donde los parámetros de entrada no modifican el estado interno del objeto 3D por lo cual su estado final (y por lo tanto también la imagen de salida renderizada) estará determinado exclusivamente por el último parámetro de entrada. Este es el caso de sistemas en los cuales los parámetros de entrada son, por ejemplo, posición del observador o iluminación.

- Determinista global: En los cuales los parámetros de entrada modifican el estado interno del objeto 3D a renderizar, por lo cual su estado final estará determinado no solo por los últimos parámetros de entrada sino también por el estado interno que, al ser un sistema determinista, es equivalente a todos los parámetros de entrada previos con la información del momento del tiempo de ser aplicados. Este es el caso de sistemas en las cuales existen, además de los anteriormente mencionados, parámetros de entrada tales como fuerza sobre un objeto.

El sistema propuesto en la próxima sección puede funcionar en ambas modalidades.

1.4. Trabajos relacionados

El protocolo presentado en el siguiente trabajo, si bien guarda relación con otros protocolos de flujo multimedia, está más fuertemente relacionado con protocolos destinados al cómputo paralelo. Entre ellos, destaca el protocolo de comunicaciones utilizado en *Message Passing Interfase (*MPI) [Gropp, 1999], el cual es empleado en computación concurrente aportando la sincronización entre procesos mediante el paso de mensajes. El mismo supone una valiosa contribución a la computación paralela. Sin embargo, está diseñado para aplicaciones en las cuales generalmente el resultado es un proceso computacional que se desea que se procese rápidamente mediante el uso de múltiples nodos. Sin embargo raramente es requerimiento que el resultado de dicho proceso sea para uso en tiempo real. MPI generalmente se implementa sobre la capa de transporte TCP, obteniendo todas las ventajas del mismo, pero al tratarse de un protocolo bidireccional no es adecuado para los casos de procesamiento destinados a tiempo real.

Por ello es apropiada la propuesta de un protocolo, que al igual que el utilizado en MPI, aporte sincronización entre procesos, pero sin la necesidad de múltiples comunicaciones ni comunicación de respuesta. Dicha propuesta parte de la determinación de que un único nodo en el sistema contendrá el estado del sistema y que la propagación del estado será unidireccional y compacta a cada nodo de procesamiento. De esta forma resulta en un sistema con la menor latencia posible para el caso específico de generación de flujos multimedia en tiempo real. Además se obtiene la ventaja adicional de funcionar en condiciones de saturación del canal de comunicaciones.

Por lo tanto, en la próxima sección abordaremos este tema con más detalle, aportando una propuesta de sistema de renderizado, a partir del cual planteamos el protocolo planteado en el trabajo.

2. Sistema de Renderización Distribuida en Tiempo Real

El sistema propuesto está compuesto por 3 tipos de nodos, cada uno de los cuales tiene un rol definido. Los roles que puede asumir un nodo en el sistema son:

- *Nodo Administrador (NA):* Es el nodo que recibe y administra los parámetros de entrada recibidos desde un dispositivo de entrada para transmitirlas por red a los *Nodos Procesador*. Adicionalmente debe anunciar a los *Nodos Procesador* qué nodo o *Nodos Integrador* les corresponde. Opcionalmente puede determinar, o no, que procesa cada *Nodo Procesador*.
- *Nodo Procesador (NP):* Es el nodo que realiza el renderizado y envía el trabajo finalizado por red al *Nodo Integrador*. Puede asumir, en el caso de que el *NA* no lo haga, la función de auto asignarse la tarea a procesar.
- *Nodo Integrador (NI):* Es el nodo que recibe todas las tareas procesadas y las integra pudiendo ser el encargado también de realizar la visualización en tiempo real de la escena renderizada. Pueden existir múltiples *NIs* pero existe una limitación, cada *NI* tendrá asignado determinados segmentos de la imagen que no podrán estar asignados simultáneamente a otros *NIs*.

Considerando los distintos tipos de nodos, el sistema de renderización distribuido consta de un *NA* para coordinar las tareas a realizar por los demás nodos, el

cual puede estar separado del resto del sistema en una red alejada permitiendo el control del sistema remotamente; varios *NPs*, quienes realizan la renderización propiamente dicha, y uno o varios *NIs* responsables de reunir y realizar la visualización en tiempo real del volumen. Los NPs y NIs deben estar ubicados físicamente cercanos para su conexión a través de una red de alto rendimiento y baja latencia. Múltiples *NAs* pueden ser soportados para funcionamiento multi-usuario o multi-vista [Hubner 2007]. Además múltiples *NI* son soportados para incrementar la resolución de salida mediante la utilización de muros de pantallas [Johnson 2012][Jiang 2015] o para incrementar el ancho de banda de tráfico de datos de salida unicast cuando el limitante es la interfaz del *NI*.

En la siguiente sección analizamos el modelo de comunicación propuesto para poder gestionar este sistema.

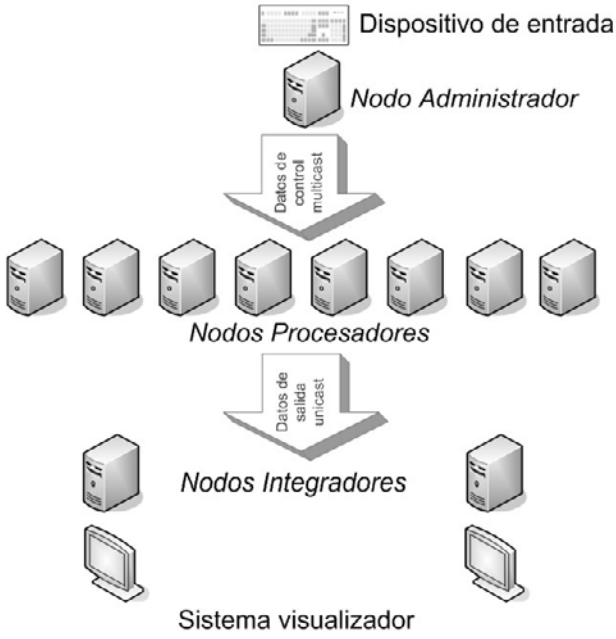

Figura 1. Esquema del sistema

2.1. Modelo de Comunicación Propuesto

Los nodos en el sistema interactúan entre sí de una manera determinada (ver figura 1), estableciendo de este modo dos tipos de comunicaciones, las cuales son:

- Información de control : Esta comunicación tiene lugar desde el *NA* a los *NPs* para enviar la información de los parámetros de entrada e información de pertenencia de los segmentos de tareas que le permite conocer a los *NPs* qué segmento de tarea debe procesar y a cuál *NI* se debe enviar cada segmento de tarea procesada. Dado los escasos requisitos necesarios de ancho de banda, esta información puede ser transferida a través de enlaces WAN a redes remotas. La utilización de multicast disminuye los efectos que la latencia de los enlaces WAN pueda ocasionar en el funcionamiento del sistema.

- Información de salida: Esta comunicación se lleva a cabo entre los *NPs* y los *NIs*. Cada una de ellas contiene la información del flujo multimedia distribuido, el cual está compuesto por los sucesivos cuadros en el tiempo generados por cada uno de los *NPs*. Los requisitos de gran ancho de banda y baja

latencia requieren de un enlace LAN exclusivo para la comunicación de salida entre los *NPs* y los *NIs*.

Como se indicó anteriormente, el presente trabajo está orientado exclusivamente al protocolo de la información de control. En las siguientes secciones se detallan las principales características y el modo de funcionamiento del modelo de comunicación.

2.2. Características Destacadas del Modelo de Comunicación

El modelo de comunicación diseñado tiene las siguientes características:

- Segmentado Jerárquico: El protocolo divide el flujo multimedia a procesar en segmentos en el tiempo y en el espacio para facilitar su procesamiento distribuido. Definimos dos niveles de segmentación, el primero dentro del flujo multimedia, cada bloque de información multimedia que comparte la misma etiqueta de tiempo la denominamos bloque de datos, *Data Block (DB)*. Luego se realiza la división espacial de dicho *DB* a lo que denominamos segmento espacial de datos, *Spatial Data Segment (SDS)*. Esta característica es similar a la utilizada por modelos de programación para GPUs [Kirk 2010] [Perez-Monte 2014]. Un ejemplo común de *DB* es un cuadro o imagen en una sucesión de cuadros de un flujo de video multimedia, mientras que un *SDS* es una porción o subdivisión espacial de dicho cuadro o imagen renderizada.
- Mejor esfuerzo: El protocolo al igual que el sistema en su conjunto deben tener una filosofía del "mejor esfuerzo" en el sentido de que ante limitaciones reales tales como ancho de banda de enlace, *NPs* con diferente potencia de cómputo, nodos ocupados, etc., el sistema debe hacer el mejor esfuerzo posible para llevar a cabo la tarea. Por esa razón que el protocolo elegido es IPv6-UDP, el cual es orientado a datagramas siguiendo también un modelo de entrega de mejor esfuerzo.
- Tolerante a saturación de enlace: De la mano del concepto anterior, el protocolo debe ser capaz de aprovechar al máximo el ancho de banda de red y ante la saturación del mismo debe ser capaz de continuar en operación normal. La pérdida de información no repercute en el sistema ya que cada paquete de datos o datagrama incluye toda la información necesaria sobre el estado del sistema.
- Energéticamente eficiente: El procesamiento de información insume un consumo de energía considerable sobre los *NPs* y como el sistema de comunicaciones realiza el máximo esfuerzo por transportar los datos de salida pero no siempre puede lograrlo, es conveniente procesar solo la cantidad de datos que el enlace entre los *NPs* y los *NIs* sea capaz de transportar.
- Tolerante a fallas de *NP* o *NI*: Cuando la cantidad de *NPs* crece, las posibilidades de que al menos un nodo falle se incrementa, es por ello que, si la política de renderización lo acompaña, el protocolo está preparado para tolerar la falla de un nodo y que el sistema continúe en operación.
- Sin Estado Interno en *NP* : El estado completo del sistema es enviado en la información de control incluso si los parámetros de entrada modifican el estado interno, como es el caso de sistemas deterministas globales.

Estas características de diseño del modelo de comunicación determinan tanto el

modo de funcionamiento como el diseño del protocolo, ambos descriptos a continuación.

2.3. Modo de Funcionamiento

El modo de funcionamiento a través de tiempo consta de tres etapas en donde los estados generales del sistema compuestos por los parámetros de entrada y sus correspondientes parámetros de tiempo son administrados por el *NA*.

En una primera etapa el *NA* debe obtener información sobre los identificadores únicos de *NP* y *NI*. Dicho método, tanto de identificación individual, grupal y de descubrimiento no está cubierto por el presente trabajo y se espera utilizar protocolos de comunicación existentes para dicha etapa.

En una segunda etapa, en la cual está definido el protocolo, el *NA* envía, por multicast a todos los *NP,* los datos de control que constan de:

- Especificación de presencia obligatoria de cómo generar el flujo multimedia.
- Especificación opcional de qué *NP* deben realizar procesamiento por cada *SDS*.
- Especificación opcional de los *NI* a los cuales enviar el procesamiento realizado por los *NPs*.
- Especificación de presencia obligatoria de los parámetros de entrada capturados de un dispositivo de entrada con información del tiempo al momento de la captura.

Así los *NPs* reciben del *NA* qué *SDS* procesar y a qué *NI* enviar el resultado o, por el contrario, determinan internamente ambas decisiones. El envío se realiza periódicamente con lapsos de tiempo inferiores al tiempo de refresco del sistema visualizador asegurando así que siempre los *NPs* tengan tarea para realizar. Se puede considerar que el *NA* gestiona la creación de un flujo multimedia virtual, de forma tal que los *NPs* generan un flujo multimedia real de menor rendimiento de acuerdo a la capacidad de cómputo global permitida. Cada *NP* recibe los datos de control, eligiendo el más actual posible y descartando el resto, y a partir del cual realiza el procesamiento adecuado de cada *SDS* de cada *DB*.

En una tercera etapa, no cubierta aún en el presente protocolo, los *NPs* envían los SDSs procesados por tráfico unicast a él o los *NIs* correspondientes. El envío de esta información se realiza tan rápido como cada *NP* es capaz de hacerlo, siendo el límite impuesto por el procesamiento de los *NPs* o por el ancho de banda de la red de entrada de los *NIs*. Los *NIs* tienen como función reunir el flujo multimedia distribuido y finalmente visualizarlo en un sistema visualizador compuesto de tantas pantallas como *NIs* existan o reenviarlo a un único nodo visualizador local o remoto para que realice la visualización en un único dispositivo de salida.

2.4. Principios de Diseño del Protocolo.

El protocolo debe contar con determinados principios de diseño, las cuales han sido determinados para un adecuado funcionamiento y flexibilidad en el uso:

- Existirá un identificador único de tamaño compacto para identificar a :

    - Cada uno de los *SDSs* dentro de un *DB*.
    - Grupos de *SDSs* dentro de un *DB* incluido un grupo todos los *SDSs* de un *DB* y ningún *SDS* del *DB*.
    - Cada uno de los *NPs*.
    - Grupos de *NPs* incluido un grupo todos los *NPs*.
    - Cada uno de los *NIs*.
- Los *NPs* pueden procesar múltiples *SDSs* de cada *DB*. Incluso más de un *NP* pueden procesar el mismo *SDS* de un mismo *DB* (de forma diferente o no). Métodos de renderización iterativos [Perez-Monte 2013] o tolerantes a fallas nodales pueden justificar este modo de funcionamiento.
- Diferentes *SDSs* de un mismo *DB* pueden ser enviados a diferentes *NIs* pero un *SDS* de un *DB*, si bien puede ser procesado por más de un *NP*, solo puede ser enviado a un único *NI*.
- Por defecto todos los *NPs* no especificados procesan todos los *SDSs* de cada *DB*. En el caso de que el *NA* lo especifique, *NPs* especificados solo procesan los *SDSs* que son explicitamente especificados.
- Por defecto los *NPs* no envían las tareas de los *SDSs* asignados a ningún *NI*, por lo cual el *NA* debe indicar al menos un *NI* por cada *SDS* procesado para que el sistema funcione adecuadamente ya que los *NPs* interrumpirán el procesamiento en caso de no poder realizar el envío de dicho procesamiento a los *NIs*.
- El *NA* podrá especificar que:
    - Un *SDS* específico o un grupo de *SDSs* específico sea procesado por un *NP* o grupo de *NPs*.
    - Un *SDS* específico ya procesado o un grupo de *SDSs* específico ya procesados sea enviado a un *NI* especificado. No deberán especificarse grupos de *NIs*.
    - Todos los *SDSs* o grupos de *SDSs* ya procesados por especificados *NPs* o grupos de *NPs* sean enviado a un *NI* especificado.
    - Todos los *SDSs* o grupos de *SDSs* ya procesados y enviados a especificados *NIs* sea procesados anteriormente por especificados *NPs* o grupos de *NPs*.
    - Todos los *SDSs* o grupos de *SDSs* ya procesados por especificados *NPs* o grupos de *NPs* sean procesados también por otro nodo o grupo de *NPs* especificados.
- En caso de que un *SDS* sea especificado para procesarse por más de un *NP*, será procesado por cada uno de ellos y por todos los *NPs* no especificados. Esto no se aplica a grupos de *NPs*. Esta característica se denomina redundancia de procesamiento automática.
- En el caso de que un *SDS* procesado sea especificado para enviar a más

de un *NI*, será enviado solo al de mayor privilegio. En caso de no poder hacerlo, se enviará al subsiguiente en privilegio de los especificados. Esta característica se denomina redundancia especificada de integración.

- Si un *NP* es asignado más de una vez a diferentes *SDSs*, estará asignado al conjunto de los *SDSs* de todas las asignaciones (operación OR).

Todas los principios y características del modelo de comunicaciones descriptos previamente permiten la especificación de una propuesta de protocolo como se describe en la próxima sección.

3. Protocolo SDCP

El protocolo diseñado es denominado Streaming Distributed Control Protocol (SDCP) y se basó en trabajos previos para los cuales se incorporaron los principios considerados en la sección previa. A lo largo de la presente sección describiremos los protocolos de capa inferior requeridos para su funcionamiento, la descripción del protocolo y los dos modos fundamentales de funcionamiento.

3.1. Arquitectura de la Pila de Protocolos

La arquitectura de la pila de protocolos que se utiliza para cumplir con los objetivos precedentes, es la siguiente:

- Capa de enlace de datos: En el caso de los enlaces entre los *NPs* y *NIs* se utiliza Ethernet con control de flujo (opcional) [IEEE 1997]. En caso de tener control de flujo, se permite que ante saturación de enlace los *NPs* solo procesen lo que son capaces de enviar por la red, lo cual reduce el consumo energético. Una implementación con esta capa de enlace ha sido validada experimentalmente; se presentó además un análisis de rendimiento en un trabajo previo [Perez-Monte 2014].

- Capa de red: Protocolo IPv6 [Deering 1998], el estándar de red de alcance mundial en su versión más reciente que lo hace adecuado para computación distribuida y con multicast para reducir el tráfico de información de control.

- Capa de transporte: Protocolo UDP [Postel 1980] o UDP-Lite [Larzon 2004], implementando un protocolo con un modelo de entrega de mejor esfuerzo y tolerante a saturación de enlace. En el caso de utilizar UDP-Lite se exige que el checksum abarque al menos los primeros 256 bits del protocolo correspondientes al encabezado obligatorio.

De esta forma SDCP será un protocolo definido en la pila de protocolos en capa de aplicación. A continuación se detallará la estructura y descripción del protocolo.

3.2. Descripción de SDCP

En los sistemas de renderizado descriptos se distinguen dos tipos de comunicación, las cuales son Información de Control Multicast e Información de Salida Unicast. SDCP describe la primera.

SDCP está constituido por datagramas que tienen como función principal el envío de los datos de control de entrada. Incluyen 2 cabeceras, una principal y la

siguiente opcional. La estructura es la siguiente:

Tabla 1. Protocolo de Control

Tipo de Estructura	**Campos**	**Descripción**
Cabecera IPv6 + Cabeceras opciones	Definidos por protocolo	Definidos por el protocolo según RFC 2460
	Dir. IPv6 de destino	Dirección IPv6 de destino multicast del grupo de los NP según RFC 3306.
Cabecera UDP	Definidos por protocolo	Definidos por el protocolo según RFC 768.
Cabecera de control obligatoria	Control Data type	8 bits. Tipo de control.
	M	1 bit. Bit de modo de control. Instantaneo o Historial.
	RD	3 bits. Reservado para uso futuro.
	Streaming Generator Id	20 bits. Identificador de fuente generadora (modelo 3D) de flujo multimedia.
	Var DB Counter	32 bits . Campo de contador de número de variaciones de payload enviado.
	SDCP Counter	32 bits. Campo de número de paquete enviado.
	Timestamp	64 bits. Marca de tiempo de los datos de control.
	DB type	16 bits. Descriptor de tipo de DB.
	DB size	32 bits. En múltiplos de cantidad de SDSs.
	SDS size	32 bits. En cantidad de pixels vertical y horizontal.
	Next Header Counter	16 bits. Número de cabeceras opcionales. Equivalente a tamaño de cabecera opcional en múltiplos de 16 octetos.
Optional Header Cabecera opcional de asignación de recursos.	First Assigned work	64 bits. Número de identificación de tarea asignada.
	First Resource	64 bits. Número de identificador de recurso a asignarle tareas.
	Next Assigned work	128 bits. Tantos como lo indique Next Header Counter.
	Next Resource	
Payload: Cuerpo con datos de control.	Last data control	Tamaño especificado por tipo de control . Datos de control correspondientes a la marca de tiempo actual del encabezado. Es un campo obligatorio.
	Historical timestamp	Tamaño especificado por tipo de control + 64 bits. Solo si Modo historial activado . Tantos como lo indique Var DB Counter
	Historical data control	

- Cabecera de control obligatoria: Posee la información obligatoria general.

- Cabecera de control opcional: Información opcional para anunciar a los *NPs*, dicha información puede ser sobre los propios *NPs* (que *SDS* o grupo de *SDSs* debe procesar cada uno) o sobre los *NIs* (que *SDS* o grupo de *SDSs* tiene asignado cada *NI* recibir).
- Cuerpo del paquete: Información obligatoria necesaria para la generación del flujo multimedia por parte de los *NPs*.

El datagrama de la información de control se muestra en la tabla (ver tabla 1).

A partir de la estructura y descripción realizada, a continuación presentamos los modos de renderización que definen el contenido y estructura del cuerpo del protocolo.

3.3. Modo de Renderización

El objetivo fundamental del protocolo es permitir enviar los parámetros de entrada de renderización conocidas por el *NA* a todos los *NP*.

Cuando los parámetros de renderización no generan cambios internos en el modelo 3D cada *NP* puede renderizar la escena simplemente conociendo el último Dato de Control conocido y el sistema funcionará en Modo Instantáneo indicado en el bit M del encabezado principal y correspondiente a un sistema de renderización determinista externo. Cuando los parámetros de renderización generan cambios internos en el modelo 3D, cada *NP* sin estado interno solo puede renderizar adecuadamente la escena si conoce todos los datos de entrada desde el inicio de generación del flujo multimedia y el momento del tiempo en que cada uno de ellos fue generado. En este caso el sistema funcionará en Modo histórico y será así indicado en el bit M del encabezado principal. Por otro lado, en el cuerpo del paquete existirán tantos grupos de datos de entrada con su correspondiente marca de tiempo como lo indique el campo Var DB counter del encabezado principal. De esta forma el sistema de renderización será determinista global.

Finalmente, a continuación, se describe el contenido de los encabezados opcionales que definen el funcionamiento del sistema de acuerdo a las necesidades del *NA*.

3.4. Modo de Asignación de Trabajo a los Recursos

Los modos de asignación de tareas (en donde una tarea puede ser tanto procesar un *SDS* por un *NP* o que un *NI* reciba un determinado *SDS* ya procesado) están determinados por la información contenida en el encabezado opcional.

Los *NPs* actúan en un modo libre, por defecto todos los *NPs* pueden procesan todos los SDSs (la política de elección es internamente elegida por dichos nodos) y más de uno puede hacerlo simultáneamente (a menos que se explicite). Por el contrario los *NIs* funcionan en un modo restringido, por defecto ningún *NP* envía SDSs a ningún *NI* a menos que se explicite lo contrario. Si se asignan los mismos *SDSs* procesados a más de un *NI*, los *NPs* envían éstos al de mayor prioridad (a menos que falle, en tal caso se envía al siguiente en prioridad). Los modos de asignación son los expresados en la tabla 2.

Tabla 2. Modos de asignación de recursos

Modo de asignación	Identificador de trabajo	Identificador de Recurso
Asig. para procesamiento del SDS por NP	Id. de un SDS	Id. de un NP
Asig. para procesamiento del grupo de SDSs por NP	Id. de grupo de SDSs	Id. de un NP
Asig. para procesamiento de SDS simultáneamente por todo el grupo de NP	Id. de un SDS	Id. de un grupo de NPs
Asig. para procesamiento de todo el grupo de SDSs simultáneamente por todo el grupo de NP	Id. de un grupo de SDSs	Id. de un grupo de NP
Asig. de envío de SDS procesado a NI	Id. de un SDS	Id. de un NI
Asig. de envío de un grupo de SDSs procesados a un NI	Id. de grupo de SDSs	Id. de un NI
Asig. de los SDS procesados por el NP 1 para que sean procesados también por el NP 2	Id. de NP 1	Id. de NP 2
Asig. de todos los SDSs procesados por el grupo NP 1 para que sean procesados también por grupo NP 2	Id. de grupo de NP 1	Id. de grupo de NP 2
Asig. de los SDSs procesados por NP para que sean enviados al NI	Id. de NP	Id. de NI
Asig. de todos los SDSs procesados por el grupo de NP para que todos sean enviados al NI	Id. de grupo de NP	Id. de NI
Asig. de todos los SDSs que serán enviados al NI para que sean procesados previamente por el NP	Id. de NI	Id. de NP
Asig. de todos los SDSs que serán enviados al NI para que sean procesados previamente por todos los Nodos del grupo de NP	Id. de NI	Id. de grupo de NP
Prohibido	Id. de NI 1	Id. de NI 2
Prohibido	Id. de SDS 1	Id. de SDS 2
Reservado para Uso Futuro	Id. de un SDS	Id. de grupo de SDSs
Reservado para Uso Futuro	Id. de grupo 1 de SDSs	Id. de grupo 2 de SDSs
Sin efecto	Id.	Mismo Id.
Comportamiento opuesto al por defecto. Dicho NP o grupo de NP no realiza procesamiento alguno	Id. de Grupo de ningún SDS	Id. de NP o Grupo de NP presente una sola vez.
Comportamiento opuesto al por defecto. Dicho NI recibe todos los SDSs procesados	Id. de Grupo de todos los SDSs.	Id. de NI presente en posición de máximo privilegio.
Comportamiento por defecto. Asignación de todos los SDSs para procesar por dicho NP	-	Un id. de NP no está presente en este campo y además no está presente ningún id de grupo al cual dicho NP pertenece
Comportamiento por defecto. Asignación de ningún SDS a enviar a dicho NI	-	Un id. de NI no está presente en este campo
Asignación del SDS o grupo de SDSs al NI de más prioridad	SDS o grupo de SDSs a través de cualquier tipo de Id.	Más de un NI presente en este campo
Asignación del SDS o grupo de SDSs al NI siguiente en prioridad	SDS o grupo de SDSs a través de cualquier tipo de Id.	Más de un NI presente en este campo y el de más prioridad falla (NPs detectan NI inalcanzable)
Descartar	Id. desconocido	Cualquier Id válido
Descartar	Cualquier Id. válido	Id desconocido

De esta forma y, para finalizar, la utilización de segmentación de cómputo gracias al uso de *SDSs* permite una gestión adecuada de los recursos gracias a la implementación de identificadores únicos de 64 bits que permiten representar inequívocamente tanto recursos como segmentos de cómputo. Además el uso de 64 bits representa una longitud de código lo suficientemente grande como para permitir un seudo-checksum implícito,

ya que los errores de transmisión tendrán la probabilidad varios órdenes de magnitud superior de generar un identificador desconocido que transformarse en otro identificador válido.

4. Conclusiones y Trabajos Futuros

El protocolo diseñado cumple con los requisitos necesarios para el control de un flujo multimedia distribuido y permite la gestión tanto de tareas como de recursos de procesamiento. Como se indicó, el protocolo posibilita una implementación eco-eficiente, ya que ante la ausencia o daño del *NA*, el comportamiento por omisión de los *NPs* es procesamiento de todos los *SDSs* y envío a ningún *NI*, por lo cual el flujo de procesamiento se detiene por falta de destino y se evita un consumo de energía innecesario. Además permite un funcionamiento redundante resistente a fallas de *NPs*, fallas de *NAs*, fallas de *NIs* y fallas en la transmisión entre *NA* y *NP*.

El presente trabajo está orientado exclusivamente al estudio del protocolo de control de la generación multimedia distribuida para su discusión en grupos de tareas específicos de IETF. En un futuro se realizará un estudio de la asignación de grupos ya sea de *NPs* o de *SDSs* y se propondrá un protocolo para los datos de salida en el caso de no existir soluciones estándar previas.

5. Agradecimientos

Se agradece el asesoramiento y soporte de: Dept. of Bioengineering y Dept. of Biomedical and Health Information Sciences pertenecientes a University of Illinois at Chicago, Argonne National Laboratory, GridTICs de UTN-FRM, LICPaD de UTN-FRM y LIDIC de UNSL.

El trabajo es sostenido económicamente gracias al financiamiento de los proyectos 25J084 “SARA Operation” y PICT2010/29 “Procesamiento para visualización utilizando algoritmos paralelos en GPU y distribuidos en red” ambos de UTN-FRM, PROICO-30310 de UNSL y de otras fuentes de financiamiento de grupo GridTICs.

Finalmente, se agradece especialmente el aporte económico de UTN con una beca de doctorado y el excelente soporte académico de la carrera de doctorado de UNSL.

Referencias

Cox, M. (1995) “Algorithms for Parallel Rendering”. PhD thesis, Department of Computer Science, Princeton University.

Deering, S. and Hinden, R. (1998) “Internet Protocol, Version 6 (Ipv6) Specification” IETF Standards Tracklo dos Santos, and Dirk Reiners (Eds.). Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 29-36. DOI=http://dx.doi.org/10.2312/EGPGV/EGPGV07/029-036ywhere: an open cloud gaming system. In *Proceedings of th*

Gropp, W., Lusk, E., & Skjellum, A. (1999). “Using MPI: portable parallel programming with the message-passing interface” (Vol. 1). MIT press.

Hubner, T. and Pajarola, R. (2007) “Single-pass multi-view volume rendering”. In: Proceedings of International Conference Computer Graphics and Visualization.

IEEE Standards for Local and Metropolitan Area Networks: (1997) “Supplements CSMA/CD Access Method and Physical Layer Specifications” - 100BASE-T2, *I*EEE Std 802.3x-1997 and IEEE Std 802.3y-1997 (Supplement to ISO/IEC 8802-3: 1996; ANSI/IEEE Std 802.3, 1996 Edition), vol., no., pp.0_1,324, 1997”.

IETF Datatracker: ART (2016) - “Applications and Real-Time Area” https://datatracker.ietf.org/wg/#art.

Jiang, J., Hereld, M., Insley, J., Papka, M., Rizzi, S. and Uram, T. (2015) “Streaming ultra high resolution images to large tiled display at nearly interactive frame rate with vl3”. In: Large Data Analysis and Visualization (LDAV), 2015 IEEE 5th Symposium on. 2015, p. 133–4. doi: 358 10.1109/LDAV.2015.7348084.

Johnson, G., Abram, G., Westing, B., Navr’til, P. and Gaither, K..(2012), “Displaycluster: An interactive visualization environment for tiled displays” . In: Cluster Computing (CLUSTER), 2012 IEEE International Conference on. 2012, p. 239–47. doi:10.1109/CLUSTER.2012.78.

Kirk, D. and Hwu, W. (2010) “Programming Massively Parallel Processors, A Hands on Approach” , Elsevier, Morgan Kaufmann.

Larzon, L., Degermark, M., Pink, S., Jonsson, L., Ed., and Fairhurst, G. Ed. (2004), "The Lightweight User Datagram Protocol (UDP-Lite)", RFC 3828, DOI 10.17487/RFC3828, IETF Standard Protocol.

Marchesin, S., Mongenet, C., and Dischler, J. (2008). Multi-gpu sort-last volume visualization. In Eurographics Symposium on Parallel Graphics and Visualization (EGPGV08).

Molnar, S., Cox, M., Ellsworth, D., and Fuchs, H. (1994) “A Sorting Classification of Parallel Rendering.” IEEE Computer Graphics and Algorithms, pages 23-32.

Moloney B., Ament M., Weiskopf D. and Moller T. (2011) "Sort-First Parallel Volume Rendering," in IEEE Transactions on Visualization and Computer Graphics, vol. 17, no. 8, pp. 1164-1177, Aug. 2011.

Monfort, J. and Grossman, M. (2009) “Scaling of 3D Game Engine Workloads on Modern Multi-GPU Systems” - HPG-09.

Perez-Monte, C., Piccoli, F., Luciano, C., Rizzi, S., Bianchini, G. and Scutari, P. (2013) “Estimation of volume rendering efficiency with GPU in a parallel distributed environment”. Procedia Computer Science 2013;18(0):1402 –11. ICCS 2013

Perez-Monte, C., Mercado, G. , Taffernaberry, C. and Piccoli, F. (2014) - “Protocolo de comunicaciones para renderización distribuida en tiempo real” CSBC 2014 - IWPIETF LAC - 28 al 31 de julio del 2014 – Brasilia – Brasil.

Postel, J., (1980) “User Datagram Protocol”, STD 6 , RFC 768, DOI 10.17487/RFC0768, IETF Standard Protocol.

Schwarz, N. and Leigh, J. (2010) “Distributed Volume Rendering for Scalable High-resolution Display Arrays”, Grapp 2010