



166166

1

9904118-8
18 February 2004

TIME BASED MULTIMEDIA OBJECTS STREAMING APPARATUS.

FIELD OF THE INVENTION

5 This invention relates to the delivery of scalable multimedia objects to meet real-time applications on a heterogeneous network. These scalable multimedia objects may contain synchronised audio, video and data information for interactive, communicative and broadcast applications over a wide area network. The present invention relates to a multimedia objects streaming method based on network resource availability at each
10 network component. It allows real-time end to end delivery of multimedia objects with minimal effect on the presentation quality of the multimedia objects over the network with dynamically changing network resource parameters, such as varying bandwidth, delay and latency jitters. This invention leads to methods and apparatus that allow fast high-level application data context switching and routing at the network layer.

15 BACKGROUND

The current delivery of real-time data depends very much on the feed back channel from the receiver to the sender to provide a report on received data transmission statistics. Based on this feedback report, the senders will take appropriate actions on the transmission of subsequent data to the user. The actions taken by the senders may require
20 reservation of more network resources for end-to-end data delivery and re-transmission of lost data. Re-transmission of lost data requires the receiver end to report specifically the amount of lost data encountered before the entire data can be of use to the applications at the receiver end. For a large system where there are a large number of servers and clients, the report messages between the terminals utilise a considerable amount of bandwidth.

25 In both connection and connectionless data network, data packets at the lower layers, such as the network layer, treat data as transport for data loads, differentiated only by the priority or the time to live parameters. This distinction made between data packets at the network layer is unable to relate the coupling nature of the application layer data fragmented into data packets for network layer delivery. Some of these data packet
30 parameters used to distinguish the various class of service between data packets are listed



ACTION

in the IETF (Internet Engineering Task Force) RFC 791 on Internet Protocol version 4 and IETF's RFC 2460 on Internet Protocol version 6. In the current specifications for Differentiated Service as mentioned in IETF's RFC 2474, RFC 2475, RFC 2597 and RFC 2958, it only provides a means to provide data packet marking. IP data packets with
5 special marking allow the network components, such as routers or switches, to treat the IP data packet based on its different quality of Service fields, DSCP, as the indicator on how data flow should be determined. The re-transmission and dropping of data packets that belong to the specific stream is performed at the IP packet level without any knowledge of the application data. Data packet differentiation is only performed at the packet level and
10 selective dropping of a group of data packet is almost impossible if this process is to be performed at the network level.

Without additional information from the application software, the data packet that made it though the network may only consist of only part of the data the application software needs. With the re-transmission of the missing data that finally arrived at the receiver end,
15 the strict real-time requirement in most instances cannot be met. A data object that arrived not adhering to the real-time requirement is said to be of no presentation value, because the time for decoding the piece of information has already elapsed. As a result, network resources are being used in delivering real-time data of no value to the end user. In addition, this non-presentation value group of data packets may cause unnecessary
20 network congestion at the already strained intermediate network nodes. In many instances the congestion of network traffic may seem to be prolonged and in a worst case scenario the traffic congestion may spread to a wider area.

In most real-time data streaming across a packet data network, the decoding rate of such a data rate is only made known to the end devices, for instance, the server and the end client.
25 In many instances, the server can only deliver data at the rate based on the decoding rate of the data and the report sent from the client giving statistics of the data received. In some implementations, a large data buffer or an arbitrary size of data buffer need to be pre-allocated at the client side before the streaming of data from the server to the client can begin. In the scenario where a wide area network is constructed based on a number of
30 intermediate network nodes such as routers and switches, these have no information on the type of data and how the data should be treated for end-to-end delivery. The intermediate

nodes have no real-time streaming knowledge of the network packets arriving and leaving the nodes. In the event of network congestion at one of the intermediate nodes, there is no means to deliver or re-schedule the delivery of network data packets that make sense from the view of the application data. In most cases, only a partial object data is successfully
5 delivered to the end terminals, and re-transmission of lost data may be performed automatically by the network layer.

SUMMARY

According to one aspect of the invention, there is provided a system for interconnecting object streaming apparatus into an existing network interconnection scheme, to control
10 data traffic generated by multimedia object streams and to ensure timely end-to-end delivery of multimedia objects, said system comprising: an Ingress Gateway to control entry of data packets into the network; a Content Traffic Exchange Hop attached to a legacy switching or routing network component; and an Egress Gateway to control the exit of data packets from the network, and to stream content to end-users of different network
15 access and computational power profiles.

According to a second aspect of the invention, there is provided a method for delivering synchronised time stamped multimedia objects as payload of network data packets, said method comprising the steps of: fragmenting the data objects into one or more data packets; marking data packets containing part or a full portion of a single multimedia
20 object using a unique reference; creating a link for related fragmented network data packets originating from the same multimedia object using a object reference before transmission; stamping the data packets relating to the object reference with time reference markings the instant the data packets leave the transmission node; and attaching the precedence level of the object, based on the object reference of the multimedia object or
25 on network data packets from the said multimedia object.

According to a third aspect of the invention, there is provided a method for scheduling the transmission time of multimedia objects to meet real time delivery and achieve constant delays between inter-arrival of multimedia objects at a next network node, said method comprising the steps of; dividing a transmission time into frames based on an Object

- Decoding rate; further dividing the transmission time within each frame of a multimedia object channel into time slots, the number of time slots being equivalent to dividing the transmission rate by the multimedia object decoding time based on the assigned bandwidth for the object channel or the amount of remaining transmission time; performing an
- 5 emulated delivery time schedule based on multimedia objects with time stamps within the scheduled object delivery period or frame period; computing a scheduled object delivery period which is equivalent to the maximum object group delay period allowable; and re-scheduling an object that cannot be delivered within the time slot, by storing it in a cache memory, the object being re-scheduled to be delivered at a time slot assigned later.
- 10 According to a fourth aspect of the invention, there is provided a clocking mechanism at a host and routing node to synchronise with multimedia object streams received and to schedule transmission time of object streams at a predetermined time slot, said clocking mechanism comprising: a module to determine the arrival of a 1st data packet belonging to a object, based on a multimedia object reference; a module to compute and maintain the
- 15 time table of all the streams, based on the difference between the last sent object and the next scheduled transmission of that same object; a link list for the data packets received for all subsequent data packets belonging to the same object; and a queuing module to put data packets in assigned allotted time slots.

- According to a fifth aspect of the invention, there is provided means of avoiding network
- 20 congestion and to limit short-term bandwidth utilisation peaks by delaying the transmission of data objects by using cache memory for data objects to meet real-time decoding at an end receiver.

- According to a sixth aspect of the invention, there is provided means of moderating a peak data rate resulting in a burst of data objects at network nodes, by diverting object data to
- 25 cache memory, object data of lower precedence using an appropriate filter.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 illustrates the management of streamed data traffic from the multimedia server or any other client user terminals in the content provider domain to the receivers at the content consumer domain via the service provider domain. This figure also shows the

interconnections of content stream traffic control for the 3 domains for the management of network resources for the efficient delivery of content.

FIG. 2 shows the flowchart for creating an object reference from the input object streams. It also shows a preferred format of the object descriptor.

- 5 FIG. 3 is a block diagram of the Ingress Object Traffic Controller used in controlling the flow of real-time traffic into the network.

FIG 4 is a block diagram of the Egress Object Traffic Controller and the Exchange Traffic Controller used in the implementation of the Egress Network node and Intermediate node for controlling the flow of real-time data.

- 10 FIG. 5 is a structure of the Object Frame for each Object Decoding Rate, and the time slots within each Object Frame. It also illustrates the way the number of time slots can increase or decrease in accordance with the traffic conditions.

FIG. 6 illustrates the preferred implementation of the timer circuit of a generic object traffic controller for the egress and exchange nodes.

- 15 FIG. 7 shows the detailed block diagram of Object Packet Stream Traffic Control found in the Object Stream Scheduler sub-block of the Exchange Object Traffic Controller.

FIG.8 shows the detailed block diagram of the Time Slot Allocator found in the Object Stream Scheduler sub-block of the Exchange Object Traffic Controller.

- 20 FIG. 9 shows the flowchart of the method of scheduling object data in the time slot allocated for the Object Frame. It also shows how time slots not used for current object frame are being accumulated for future use.

DETAILED DESCRIPTION

- 25 To solve the problems in the art, the present invention allows the current network configuration to use apparatus or devices implementing traffic control of real-time streams at the Content Producer domain, the Service Provider domain and the content consumer domain; a means for intermediate network component to stream real-time data; a means to
-

provide accurate time synchronisation of time based objects between network components; a means of allowing quality of service for the data objects to be delivered over the network; a means of providing linkage for fragmented object data packet from a single data object; a means of dropping object data failing to meet real-time requirement; a
5 means to provide an architecture to allow various traffic shapers or filters to be controlled by a generic traffic stream controller; a means of dividing the real-time time frame into smaller time slots for the delivery of data objects; a means of allocating and de-allocating time-slots to meet the real-time requirement of objects and improve the network traffic conditions; a means of providing the real-time traffic controller for edge devices such as
10 ingress and egress nodes as well intermediate traffic exchange node to stream multiple object data of different decoding rate simultaneously.

This invention involves system architecture that allows the end-to-end streaming of time based objects. The input object data from the object stream entering the network is marked and disassembled by the Ingress Object Traffic Controller (Ingress Gateway).

15 In the Ingress Object Traffic Controller, a large data object is fragmented into more than one smaller network data packet as specified by the chosen network layer for delivery. Each of the network data packet are marked according to the precedence level desired by the application entities at the source of the object stream. To provide the network data packets with the linkage and real-time information, each object has an object descriptor
20 created. Based on the real-time requirement and the network resources allocated for the delivery of the object stream, the time for each Object Frame is computed and created for the delivery of time-stamped object data. The data information of the object descriptor is mapped to the header of the network data, where possible, or it could be encoded into the payload.

25 At each of the intermediate node, such as routers and switches connected with the apparatus implementing the inventive steps, the data packet received is re-assembled. The data packet is re-routed to the host using the Egress Object Traffic Controller (Egress Gateway) if the data object has reached the final destination, otherwise it is processed by the Exchange Object Traffic Controller (Content Traffic Exchange Hop) for re-
30 transmission to another network node. In the Egress or Exchange Object Traffic

Controller, the object descriptor for each stream is collected and a timer circuit is spawned for each object stream. Based on the timing information the System object frame is created dynamically and the number of time slots per object frame is computed. The time slot is used to mark the delivery time of the object data. The time schedule for the delivery of the object based on time slots is performed based on the traffic shaper or filter adopted for the stream. An object meeting the criteria of the traffic shaper is put in the Object stream queue and those object data not meeting the requirements of the traffic shaper are stored in cache memory for a generalised lower priority object data scheduling. An object decoding period is determined from the set of object stream descriptors received. One or more object decoding time slots are allotted for the object data to be delivered to meet real-time criteria in the Stream Object Queue. After completing scheduling of the object data that will expire within the object time frame in the Stream Object Queue, data objects in the Stream Object cache memory are scheduled next. After an object has been scheduled, excess time slots are saved for a future Object Frame. If there are insufficient time-slots, additional time-slots are allocated from the reserves to meet the needs. If there are insufficient time-slots for delivery, object data that expire in the next object frame are discarded.

A data object packet scheduled for re-transmission is then re-assembled based on the output network criteria and a network header is mapped accordingly. The process is repeated for the data object in the Exchange Object Traffic Controller of the intermediate nodes within the service provider domain until the final destination has been reached.

The invention provides a system and apparatus for, and methods of, delivering multimedia objects for improved streaming at the multimedia content provider, improving the quality presentation at the content consumer end and improving the network congestion at the service provider domain. The present invention is described in the following in detail with reference to a few preferred embodiments thereof, as illustrated in the accompanying drawings, introduced above. In the following descriptions, numerous details are set forth in order to provide for a thorough understanding of the present invention by one skilled in the art. In some of the explanations, some obvious details that forms part of the inventive steps are left out in order to not unnecessarily obscure the present invention.

Referring now to FIG. 1, a diagram of an exemplary network system consisting of different network functional nodes for implementing the various aspects of the present invention is illustrated. It shows an additional network functional component being added to allow traffic control and management of data generated by the users in two domains; the content producer, as marked by literal 1.1, and the content consumer marked by literal 1.10. The physical end-users with communicative applications can, in different instances, be in either the Content Consumer or Content Producer domain, depending on the flow the object stream. The entity that ensures the end-to-end delivery of object streams is the Service Content Provider, 1.5. The Service Provider domain is connected to the Content Producer via a network link that shares the same network link, 1.4, and architecture as the first connected network switching element, 1.6. For brevity, the network component marked by literal 1.6, is termed a switching element, which is a representative network component for routing of data packets or performing label switching or layer 2/3 switching. The Service Provider may contain a network of interconnected switching elements that support different network protocols and architectures. Each of the switching elements is connected to a Content Traffic Exchange Hop, 1.8, to perform the function of object data interchange and to control the flow of object traffic out of the switching element. The connectivity between the switching element and a Content Traffic Exchange Hop is via a standardised interconnection, 1.7, that allows data to be exchanged, or some other network connection. This link allows the flow of selected data streams from different end-to-end users to be controlled and managed by the Content Traffic Exchange Hop before being passed back to the switching element. In the Content Provider domain, the traffic generated by each of the server entities, 1.2, is moderated by the Ingress Gateway, 1.3, to control the traffic entering the Service Provider Domain. For brevity, the term "server" is used to describe a logical entity which is the source of content to be delivered across the network. The last switching element in the Service Provider domain is connected to the Egress Gateway, 1.11, via an interconnect, 1.9, that shares the same type of network protocol and architecture. In the Content Consumer Domain, it may consist of clients with different network access profiles. These clients can either have high bandwidth, 1.14, fixed line access, 1.12, low bandwidth, 1.15, or Mobile access, 1.13. The Egress Gateway will regulate the traffic streamed to the client and requests for the server carried out by the client.

Literal 2.1 of FIG 2 shows an exemplary descriptor used by multimedia objects before sending them to the network via the Object fragmentor. The application object descriptor provides sufficient information for the object fragmentor to enable data packets belonging to the same object to be linked. Data packet linking is necessary as, for a data object from the application, in many cases if not all cases, the object needs to be fragmented into smaller packets adapted for the network delivery layer before it can be delivered by the data carriers of the network layer. The flowchart in FIG 2 shows the method for providing time information and linking of data packets from a fragmented multimedia object as application level software streams the multimedia objects into the network. For each transmission of object data, a session is set-up with the Object fragmentor, 2.2. The object fragmentation process is then carried out based on the descriptor's decoding rate and Precedence Offset parameter, 2.3. This initialisation process allows the fragmentor to decide on the value for object marking of the payload of the network delivery data packet. Based on the object descriptor and the network delivery payload size, the number of data packets to be linked is determined, 2.4. Each of the data payload for fragmented objects is then stamped with a unique Major Object Reference number, as in 2.1, a number which the object stream uses for the entire life cycle of the session. This Major Object Reference number is incremented by one for each session created by the application with the Object fragmentor, 2.5. For each data packet created from the object data, both the Major and Minor object Reference number are stamped, either as part of the data packet payload or packet data header or a representative of header information that the network data packet carries. Stamping of the data packet and packing of fragmented data into the payload of a data packet are performed by processes marked by 2.6, 2.7 and 2.8. At the end of all network data packets marking, the network data packets are time stamped, by process 2.10, with the same time value passed by the object descriptor. The object fragmentor will synchronise with coming object data if the decoding time rate is the same based on the time value in the object descriptor of the object reference.

FIG 3 shows the block diagram of the Ingress Object Traffic Controller, which enables marking and controlling of the flow of Object data entering the network. Object data enters the Ingress Object Traffic controller with the object descriptor, 2.1, passed down from the application entities. Each object data are fragmented into smaller data packet

payloads by the Objector Fragmentor, 3.5 based on the Network payload size, as marked by 3.2. For each of the data object divided into smaller data packets, the time is marked by the timer or time value in the object descriptor, as in 2.1, by the functional block marked by 3.6. The timer derives it's time value from the data descriptor's, 2.1. The decoding rate and this value is fed into the timer block, 3.6, via the input marked by literal 3.3 if the value from the subsequent Object descriptor is the same. The timer value generated is feed into Time Slot Divider, 3.8, where time slot transmissions of Object data time slots are created, based on the bandwidth allocated for the delivery medium. Based on the time stamped on the data packets belonging to the same object, the time Slot Allocator determines in which time slot the group of data packets is to be delivered to the network. For each data packet to be delivered, the data packet are mapped to the delivery Network Payload Packetizer module, 3.11, and the time information of the data packets are sent to the Network Parameter Mapper module, 3.10, before the whole network packet is delivered to the network delivery medium, 3.12.

15

FIG 4 shows the block diagram of the apparatus for receiving the object data fragmented into one or more network data packets and the apparatus that performs the full functions of the Content Traffic Exchange Hop. Further an Egress Object Traffic Controller (Egress Gateway) consists of functional blocks marked by literal 4.5, 4.6, 4.4, 4.10 and 4.2. For each of the network data packet received, the data packet header is parsed and decoded by the Network Parameter Decoding function of the block marked by literal, 4.5. The timing information and the object reference extracted from the header are passed for further processing by the Timer Circuit, 4.4. Another source of timing information and the object reference number can be extracted from the Network Payload De-Packetizer function of the functional block marked by literal 4.6. The timing and object reference are the only 2 pieces of information that are extracted from the network packet payload if the network header of the network data packets does not carry timing information and the object reference number. Encoding of the timing information and object reference number in the payload of the data packet follows the object descriptor shown in FIG.2. The Timer Circuit, 4.4, contains a time stamped time for each of the object data received within the Object decoding frame. A handle reference is created to provide a distinction between the various object data received, based not only on the object reference number extracted from

30

the object descriptor but also the source and destination network address and also the numeral representative of the network service access point. For the timer for the object received to work accurately within each frame, the timers in block 4.4 are updated or created by timing information from the Object Stream Timer Circuit, 4.12. The timing
5 information is also passed to the Object De-Fragmentor, 4.2, for accurate construction of the data stream if the data stream received has reached the final destination. For the Object Stream received, the stream is further processed by the Object Stream Classifier, 4.10, to determine the final destination of the object stream, based on the unique handle reference. If the object stream has reached the final destination, the data packets are passed to the
10 Object De-Fragmentor, 4.2. In order for the object to be re-assembled and later fed to the buffer used for object decoding, the timing information from the Timer Circuit which keep the timers for the object stream received within one Object Decoding time frame, is required.

In order for the host network components, such as switches and routers, to perform the
15 function of the Content Exchange traffic controller, all the functional sub-blocks in the Object Stream Scheduler block, 4.7, are required. The internal clocking of the timer referenced by the unique handle reference provides all the accurate time keeping and are kept by the Object Stream Timer Circuit, 4.12. This list of timers created only contains timing information for the object stream to be re-routed to another destination. For each of
20 the streams received, the delivery time slots for each of the object data, based on each individual object stream decoding timing requirement, and the transmission time required are computed by the Time Slot Divider, 4.11. In addition, 4.11 also provides the Object packet stream traffic control, 4.8, with the object descriptor constructed from the data packet received via the Network Parameter Mapper and Decoder functional block, 4.5.
25 Based on all of the information computed for each object stream of the time slot divider, the Object Stream Traffic Control, 4.8, then decides the time required for the final delivery of the object stream. Based on the precedence of the object stream carried in the object descriptor of each object stream, the Time Slot Allocator performs the final decision as to how the object data fragmented in data packets are delivered out to the network via the
30 functional block 4.6. Timing information of the data object stream that is scheduled to be transmitted is provided by the System Time Slot Divider, 4.3. The System Time Slot Divider gathers all the information of each time slot divider assigned to each object stream

to determine the minimum object decoding time. The number of time slots for the node is determined by EQ.2 (see below). The Minimum Object Frame determined, by the System Time Slot Divider, follows the function in EQ.1 (see below). Functional block 4.3 performs and maintains the universal object frame, based for the Content Exchange Traffic
 5 Controller, based on the network resources allocated for transmission. The network resources may include bandwidth, latency and total packet group delay. Information generated by 4.3, is also synchronised with the Time Slot Allocator ,4.9, such that the same timing information can be mapped to the network data header or encoded in the payload of the network packet carried out by the functional block marked by literal 4.5.

$$10 \quad \text{Object Frame} = \text{Min}(\text{DecodingRate}_{\text{Object Stream 1}}, \text{DecodingRate}_{\text{Object Stream 2}}, \dots, \text{DecodingRate}_{\text{Object Stream N}}) \quad (\text{EQ.1})$$

To enable the exact transmission time of the Object data to be transmitted out at the Egress node or Content Traffic Exchange Hop, a time frame called the Object Frame is assigned for the Object streams to be transmitted out. The pre-allocated time period for the Object
 15 Frame is equivalent to the time for decoding one Object data as described in the Object descriptor. In the case where there is a requirement to use more or less bandwidth, the object frame period can be increased or reduced. Each Object Frame may consist of more than one time slot, depending on the bandwidth allocated for the delivery of the Object data. Depending on the amount of transmission bandwidth allocated, it may contain more
 20 than one time slot. The number of time slots per Object Frame can be computed as follows:-

$$\text{No. of Time Slots (n)} = \text{Transmission rate/Object Decoding rate} \quad (\text{EQ. 2})$$

For each Object Frame, only an integer number of time slots are allowed. In the case of a non-integer value as a result of EQ. 2, additional time slots may be allocated for each time
 25 Object Frame, if the sum of the fractional portion of time slots calculated forms a value more than one, accumulated over a number of Object Frames. The rate of additional time slots in terms of Object Frames is inversely proportional to the fractional portion of the computed time slots, n, derived from EQ.1. FIG 5 shows the organisation of the Object Frame, 5.1 and time slots, 5.2. In the case where n is a fractional number, the transmission

of the Object frame is scheduled earlier than the time frame bounded by the Object Decoding rate, as in, 5.4. Over several Object Frame periods, when the fractional portion of the time slots accumulates to more than one, one additional time slot, 5.5, would be allocated for the Object Frame marked by literal 5.4. The allocation of time slots marked
5 by 5.5 is strictly based on the needs determined by module 4.8 of the block diagram in FIG. 4. The time frame of the Object Frame is dynamically updated by the object descriptor at the rate equivalent to the arrival of the network data packet of the object data.

The preferred implementation of the timer as mentioned with respect to blocks 4.4 and 4.12 of FIG 4 and 3.6 of FIG 3 is illustrated in FIG 6. This implementation of the timer is
10 used to provide synchronisation between network components. Each time a timer instance is instantiated, a timer session is created in 6.1. An individual timer instance provides a means to insulate the time reference from other object stream object references. The timer is updated with the timing information from the object descriptor, 6.2. The time value in the object descriptor provides the reference decoding time of the object data. All object
15 references with similar major reference numbers and with the same source and destination address, with similar network service access point numeral representatives use the timer session. This allows all objects within the same object stream to follow one timing reference. As the object data from the same object stream arrives, the timer is updated with the internal time, Delta_Time, as marked by literal 6.3, with time resolution finer than the
20 time period of one time slot computed. The timer session is deleted if the object stream discontinues the object stream transmission, 6.4.

Based on the information from the System Time Slot Divider time slot information for each individual object stream, the scheduling of the data object to be streamed out to another destination is performed. Literal 7.1 marks an arbitrary number of input object
25 stream traffic entering the node to be re-transmitted to another destination network node. Information such as the stream object descriptor and the information from the output of the Time Slot Divider that provides description of the object are multiplexed by 7.2. A full scan for k object streams, as in 7.1, by the multiplexer is performed at the maximum rate of the System Object Frame generated by the System Time Slot Divider, 4.5. It can be
30 performed at a faster rate if not all the streams have objects for delivery. Besides streaming the information to the Stream Object Slot Scheduler, the multiplexer also keeps

the statistics of the object stream, as shown in the data structure below. This allows a feedback for the dynamic mechanism required by the traffic shaper or filter.

```
Object Stream Statistic {
    Max Object Size in Bytes;
5    Min Object Size in Bytes;
    Average Decoding Rate;
    Total Objects received;
}
```

In the preferred art, multiplexing sequence starts from the object stream with the least
10 decoding rate, as illustrated in 7.1, and is then followed by other precedence
classifications. FIG. 7 shows the detailed description of the Object Packet Stream Traffic
control. The Object Stream Shaping, 7.3, then performs filtering based on the object data
size, the rate of entry and the precedence of each of the objects received from one single
stream. The detail shaping algorithm can vary, depending on the rate of decoding. For the
15 data that meet the filter criteria, reference of the data object is passed to the Stream Object
Slot scheduler, 7.4. The Stream Object Scheduler then assigns the object to the time slot to
deliver the object within the decoding rate as specified in the object descriptor. The time
slot schedule information, which refers to the object delivery time, in terms of time slots,
is passed to the Time Slot Allocator. References of Data Object that did not meet the filter
20 requirement are passed to the Object Cache Control, 7.5, where information of the cache
for the object is passed to the Time Slot Allocator's Stream Object cache memory of FIG
8. The Object streams in the cache memory have object precedence of lower levels and
should not be transmitted because the capacity for the stream has reached the assigned
bandwidth according to the filter in the Object Stream Shaping information.

25 FIG. 8 shows the detailed functional diagram of the Time Slot Allocator. Literal 8.1 shows
arbitrary stream traffic received. The object frame created by the Time Slot Allocator is
based on the assigned transmission bandwidth and the minimum decoding rate determined
from the System Time Slot Divider, 4.3. The time slot schedule information from the
Object Stream Traffic Control, 4.8, determines if the data selected by the Object Stream
30 Multiplexer should put the object in the Stream Object queue, 8.3 or the Stream Object
Cache Memory, 8.5. Based on the queue information from the Stream Object Queue and

the Stream Object Cache memory, the Object is slotted into the time slot of the Object frame. Slotting of object data into the time slot is performed by the Object to Time Slot Mux, marked by literal, 8.4. Because of the finite memory size of Stream Object Cache Memory, the functional block, 8.5 may choose to discard the longest staying data objects if the memory assigned is used up. If there are free time slots after the whole object in the Stream Object Queue has been allotted, the data object in the Stream Object Cache memory is multiplexed out via the Object to Time Slot Mux, 8.4, to be delivered out to the network. The minimum Queue size for each stream should not be less than the entire Object Frame size. The total memory assigned for the object stream in both cache memory and queue memory should not exceed the following to meet the total group delay for one node;

$$\text{Memory Size (Queue \& Cache)} = \text{Group Delay} \times (\text{max. Object Size} / \text{Decoding Rate}) \text{ (EQ. 3)}$$

Depending on the amount delay to be supported, the group delay value of each node needs to be updated such that re-allocation of the memory size can be recomputed. The recomputed memory size determines the maximum group delay per hop using EQ.3.

For each of the object frames computed dynamically by the functional block, 4.3, for the system implementing the Exchange Object Traffic Controller, a new number of time slots per object is computed using the conditions set forth in EQ.1 and EQ.2. The rate of change is performed at the rate of object frame rate. FIG 9 shows the flowchart for the allocation and de-allocation of time slots being carried out in a functional block marked by module, 4.8. The flowchart assumes that the allocation of all objects based on the allotted bandwidth for the intermediate node and also the precedence criteria to be implemented accordingly for the streams to be delivered, are performed based on the preference of one skilled in the art. For each object frame, the number of time slots is computed by functional block, 9.2, meeting the criteria set forth in EQ.1. Object data delivery within the object frame is performed by 9.3, based on the data packet in the Stream Object Queue. If the objects in both queues, the Stream Object Queue and the Stream Object Cache memory, have data objects that do not meet the real-time requirements, the packet with the object reference is dropped accordingly by functional process, 9.5. Objects that meet the real-time requirement have the total time the object data packets reside in the queue less

than the decoding rate of the object stream. Schedule time (t) is defined as the total time spent in the queue and time spent if the object is delivered in the next Object Frame. If the schedule time (t) is larger than the decoding rate of the object, it represents that the data object needs to be delivered in the current object frame. If additional time-slots are
5 required within the object frame, the allocation process can check with the functional block 9.7, if there are additional time slots being saved over previous object frames. The re-allocation of additional time slots is updated by the functional block, 9.8, which keeps global time slots reserved over an arbitrary period of time. With the excess time slots, data objects in the Stream Object Cache memory are scheduled to be delivered next, performed
10 based on criteria set forth in 9.9. Assignment of data objects based on the allocated time slots is performed in 9.10. After all objects within the decoding time frame have been assigned, the remaining time slots for the object frames are saved for future object frames, as marked in literal 9.11.

This invention allows the delivery of data without the additional feedback from the
15 receiver, and thus cutting down on the complexity in the protocol needed by both end users to handle feedback reports. It also reduces the processing power needed for the client to perform reporting based on the data received from the server.

This invention is able to provide a means to deliver real-time and near real-time multimedia objects and the delivery of different decoding rates in a connectionless
20 oriented connection, accurately based on individual decoding rates. The invention further allows synchronisation of the time based object stream, such that real-time delivery specified for the object stream can be met accurately without an accurate timer being implemented by the host network node.

The present invention provides a mechanism to perform network congestion avoidance at
25 each intermediate network node, dynamically, at the rate equivalent to the smallest decoding rate from the set of decoding rates from the object streams. It also allows link information between data packets belonging to the same data object to be retained as the data is fragmented and delivered across the network.

As for real time delivery, the intermittent delay is kept at the maximum constant delay, such that end to end delay for the arrival of all data objects is kept at a constant rate over time. In most networks, component such as routers may cause the delay of each data packet to exceed the real-time requirement at different times, depending on the traffic conditions encountered. These intermittent variations in delays cause the object data arrival rate to exceed the real-time requirement in many instances.

The content data switching can be performed at the network level, thus freeing the application level from the rudimentary work of re-transmitting content object data received.

10 With the ability to couple fragmented data packets originating from a single data object, the present invention enables routers or intermediate network nodes between end-to-end users to drop data packets of data objects with no presentation value. A Data Object without presentation value is an object where the data object cannot arrive at the end user at the decoding rate. This prevents non- presentation data objects from causing network congestion, by dropping the said object data.

A Timer can improve synchronisation without the need for a central control time or network time protocol to keep synchronisation between network nodes. The present invention allows an accurate means to estimate the decoding time of each object belonging to different object streams.

20 This invention also allows a single network traffic controller to control object streams with different traffic shaper or filtering algorithms. With this invention, object stream traffic shaping can be performed accurately and simultaneously with other traffic shapers.

This invention allows existing network configurations to use apparatus implementing the present invention to control real-time traffic over wide area networks such as the Internet or a corporate wide Intranet. It also provides an infrastructure for the content provider to stream AV content to users in real-time, and the amount of data objects to be delivered to end users can vary depending on the profile of the users, on a data packet network such as the Internet. For a virtual leased line, a server can choose to stream content to its valued clients with better Quality of Service over non-valued clients.

The invention provides system architecture for apparatus implementing the inventive steps, for instance an object traffic controller to be configured and connected to any legacy network structure for end-to-end delivery of real-time data. Apparatus implementing the

5 inventive steps can provide a real-time mechanism for content switching based on timing requirements of individual data streams at a network node performing the routing and switching functions. Network congestion is avoided by discarding object data which may consist of a number of network data packets, if the real-time requirement fails to be observed due to heavy network traffic. Discarding objects failing the real-time

10 requirement allows efficient use of network resources and also provides an indicator to higher layer software that the bandwidth is insufficient for the session. It also enables the network components to operate in a collaborative effort in streaming real-time traffic through a large network. It also frees up the additional resources required at the client terminals to implement sophisticated protocol stacks, for providing feedback based on the

15 traffic received. In addition, the system allows real-time with high timing accuracy delivery across a wide area network without drastic changes to the existing network architecture. The client can receive quality content with graceful degradation when there is network congestion or when the bandwidth to the client is reduced due to heavy utilisation of network resources or server resources. On the service and content provider ends, it

20 provides an infrastructure and means to differentiate real-time services provided for the client, based on the client's profile. This invention is very useful in delivering real-time data across heterogeneous networks to light-weight clients with limited network resources and processing power such a mobile PDA or a Mobile Video Terminal.

CLAIMS

- 1) A system for interconnecting object streaming apparatus into an existing network interconnection scheme, to control data traffic generated by multimedia object streams and to ensure timely end-to-end delivery of multimedia objects, said system comprising:
- 5 (i) an Ingress Gateway to control entry of data packets into the network;
- (ii) a Content Traffic Exchange Hop attached to a legacy switching or routing network component; and
- (iii) an Egress Gateway to control the exit of data packets from the network, and to stream content to end-users of different network access and computational power profiles.
- 10
- 2) The system as in claim 1, further comprising means for attaching object streaming apparatus to a legacy network inter-linked by network components supporting existing network protocols and architecture.
- 3) The system as in claim 1, wherein the Ingress Gateway comprises:
- 15 (i) an Object Fragmentor module to break up large object data into smaller data packets suited for the payload size of the network delivery medium;
- (ii) a timer, used to perform clocking functions to emulate the decoding time of the object to be decoded by the end device;
- (iii) an object marker to provides linking of network data payloads belonging to the same data objects to be delivered;
- 20
- (iv) a time slot divider to compute the transmission time of the object data fragmented into a number of data packets;
- (v) a time slot allocator to determine the exact time of delivery;

- (vi) a Network Parameter mapper to translate the timing and the object reference of an object descriptor to the header of the network data packet or to encode them as part of network data packet's payload; and
 - (vii) a Network payload packetizer to encapsulate the fragmented object data packet into the format used for delivery at the network layer.
- 5
- 4) The system as in claim 1, wherein the Content Traffic Exchange Hop is provided at a node and comprises:
- (i) a Network Parameter Mapper and Decoder module which checks and encodes the headers of the packetised multimedia objects based on the precedence level, time stamped time of multimedia objects, the source and destination reference;
 - (ii) a Network Payload Packetizer and De-packetizer used to encapsulate and assemble the object data from the data packets of the same object received from and transmitted to the Network layer;
 - (iii) an Object Stream Scheduler to register accurately the decoding time for each of the data object streams received and to schedule the delivery time of streams to be routed out of the node;
 - (iv) a Data packets transmission scheduler related to a time stamped multimedia object if transmission time allows a single multimedia object to be sent out in its entirety
 - (v) a Time Slot re-scheduler for re-allocating the sending time of an object to an earlier time slot if it fails to meet the real time requirement of the object stream;
 - (vi) a System Time Slot divider for synchronising all object streams for re-transmission or re-routing; and
 - (vii) a Time Slot Allocator for the exact delivery time of an object stream leaving the node.
- 10
- 15
- 20
- 25
-

- 5) The system as in claim 1, wherein the Egress Gateway comprises:
- 5 (i) a Network Parameter Mapper and Decoder module which checks and encodes the headers of the packetised multimedia objects based on the precedence level, time stamped time of multimedia objects, the source and destination reference;
 - (ii) a Network Payload Packetizer and De-packetizer used to encapsulate and assemble the object data from the data packets of the same object received from and transmitted to the Network layer;
 - 10 (iii) an Object Stream Classifier to determine the final destination of the object stream, based on a unique handle reference;
 - (iv) a Timer Circuit and Packet Time Decoder for containing a time stamped time for each of the object data received within an Object decoding frame; and
 - 15 (v) an Object De-Fragmentor to reconstruct the data stream from the data packets of the same object.
- 6) A method for delivering synchronised time stamped multimedia objects as payload of network data packets, said method comprising the steps of:
- (i) Fragmenting the data objects into one or more data packets;
 - 20 (ii) Marking data packets containing part or a full portion of a single multimedia object using a unique reference;
 - (iii) Creating a link for related fragmented network data packets originating from the same multimedia object using a object reference before transmission;
-

- (iv) Stamping the data packets relating to the object reference with time reference markings the instant the data packets leave the transmission node; and
 - (v) Attaching the precedence level of the object, based on the object reference of the multimedia object or on network data packets from the said multimedia object.
- 7) The method as in claim 6, further comprising the step of providing an object descriptor to allow a means for higher entities software to specify the data object linking mechanism and for synchronising multimedia objects delivered across a network in the form of fragmented network data packets.
- 8) The method as in claim 7, further comprising the step of providing means for application entities to differentiate the importance of the object data with reference to other object data originating from the same application entity.
- 9) The method as in claim 7, further comprising the step of providing means for marking the time fragmented data packets originating from a single object data, to provide a reference point and accurate timing for end-to-end synchronisation of streamed object data.
- 10) A method for scheduling the transmission time of multimedia objects to meet real time delivery and achieve constant delays between inter-arrival of multimedia objects at a next network node, said method comprising the steps of;
- (i) Dividing a transmission time into frames based on an Object Decoding rate;
 - (ii) Further dividing the transmission time within each frame of a multimedia object channel into time slots, the number of time slots being equivalent to dividing the transmission rate by the multimedia object decoding time based on the assigned bandwidth for the object channel or the amount of remaining transmission time;

- (iii) Performing an emulated delivery time schedule based on multimedia objects with time stamps within the scheduled object delivery period or frame period;
 - (iv) Computing a scheduled object delivery period which is equivalent to the maximum object group delay period allowable; and
 - (v) Re-scheduling an object that cannot be delivered within the time slot, by storing it in a cache memory, the object being re-scheduled to be delivered at a time slot assigned later.
- 11) The method as in Claim 10, further comprising the step of providing means for dynamically changing the number of time slots per frame by adapting to a changing object decoding rate of the object descriptor and changes to the transmission speed of the network link.
- 12) The method as in Claim 10, further comprising the steps of:
- (i) Computing the number of time slots available for multimedia object delivery of a current object frame;
 - (ii) Updating the transmission time slots reserved over a buffered object data in both a queue memory and cache memory of a stream connection;
 - (iii) De-allocating time slots for a subsequent frame if the transmission of intended multimedia objects cannot be delivered within an assigned frame, as determined by an object shaper; and
 - (iv) Allocating time slots from a time slot bank accumulated from previous object frames if transmission of intended multimedia objects within the specified decoding time frame cannot be delivered within the assigned object frame.
- 13) The method as in Claim 10, further comprising the step of computing a queue buffer size required for a stream queue at a host or routing node based on a maximum object group delay parameter and maximum multimedia object size.

14) The method as in claim 10, further comprising the step of providing means for streaming an elastic delay period for the arrival of a multimedia object based on a maximum object group delay parameter, said means for steaming comprising the following discrete delay models

5 (i) End to end delivery delay time for all objects is culminated at a value equivalent to the maximum object delay parameters;

(ii) Node re-transmission delay where the maximum delay of an object is culminated at a value not more than n multiples of the maximum object delay experienced at each intermediate node between the sender and receiver; and
10

(ii) Queue buffer delay introduced by a queuing algorithm at each intermediate node, where the maximum delay is based on the size of the queue buffer size divided by the minimum object size.

15 15) A clocking mechanism at a host and routing node to synchronise with multimedia object streams received and to schedule transmission time of object streams at a predetermined time slot, said clocking mechanism comprising:

(i) a module to determine the arrival of a 1st data packet belonging to a object, based on a multimedia object reference;

(ii) a module to compute and maintain the time table of all the streams, based on the difference between the last sent object and the next scheduled transmission of that same object;
20

(iii) a link list for the data packets received for all subsequent data packets belonging to the same object; and

(iv) a queuing module to put data packets in assigned allotted time slots.

25 16) The clocking mechanism as in Claim 15, further comprising means for providing synchronisation of an object stream and to limit the jitters of an object stream by using a

stream object cache memory and by multiplexing an object in the cache memory to empty time slots of an object frame.

17) A means of avoiding network congestion and to limit short-term bandwidth utilisation peaks by delaying the transmission of data objects by using cache memory for
5 data objects to meet real-time decoding at an end receiver.

18) A means of moderating a peak data rate resulting in a burst of data objects at network nodes, by diverting object data to cache memory, object data of lower precedence using an appropriate filter.

19) A means according to Claim 18, further comprising means and apparatus for
10 enabling the control of different traffic filter specifications, based on a generic method for the allocation of bandwidth by dividing a real-time delivery window in terms of decoding frames into multiple time slots.
