4

Virtual Multicast

Petr Holub and Eva Hladká
Masaryk University and CESNET z. s. p. o.
Czech Republic

1. Introduction

The ability of the Internet to facilitate collaboration leads to widespread use of various video-conferencing and more advanced collaborative environments. As a result, synchronous multimedia transmissions have become more common. Various communication patterns emerged: from many-to-many low-bandwidth streams for large scale collaboration over slow links to few-to-few extreme-bandwidth streams as seen in collaboration based on high-definition (HD) Holub et al. (2006), Jo et al. (2006) or even post-HD video Shimizu et al. (2006). These applications require Internet to become more active, the classical passive transmission service is no longer sufficient.

Multimedia streams are processed within the network, allowing, e.g., to establish a collaborating group where most members are connected to the high-bandwidth network links while a minority has rather limited connection. If the network is capable of processing—compressing, down-sampling, etc.—the data at the appropriate nodes (where the high and low throughput network links interconnect), the communication quality should not be reduced to the lowest common throughput denominator. The network must be able to support complex communication patterns and to process data internally. Robustness and failure resilience is another area, where more support at the network level is expected. While classical transport protocols like TCP support reliable data transmission, they are not appropriate for synchronous multimedia environment, where delays are unacceptable. It may be undesirable to wait for a timeout and then ask for a datagram retransmission, the network and applications themselves must be able to detect and immediately mitigate any data corruption or loss. Up to now, new requirements were served by different infrastructures tailored for a specific purpose. Nowadays, we need to merge them together in a network that uses packet transmission as its basis protocol—to do this successfully, new models, approaches, and techniques are necessary.

The theoretical model of the virtual multicast naturally follows from graph-based model of computer networks. The graph model of computer network can easily be extended to multigraphs, which allow multiple line to connect any individual nodes. Although most computer networks are bi-directional, working a semi-duplex or full duplex regime, orientation can be added for explicit description of direction of flows (multiple edges used to represent the bi-directionality). As another step, we can add labels to the edges, representing some important properties like throughput or latency of each link. Labels on nodes can denote their properties, like different capabilities, latency of passing (bridging) data between edges of the node (the internal latency), size of internal buffers, etc. We can also speak about *internal* network, which is a part of the graph without any leaf node. It is also easy to identify end—i.e. leaf—elements.

Such a model is appropriate to study most usual flow patterns in contemporary computer networks, namely the sender–receiver one. In this case, we have one node sending and exactly one node receiving a particular data flow. The basic network problem is finding a route between the communicating nodes, additional constraint is to guarantee available bandwidth and eventually other properties like overall latency or jitter. The route is usually the one composed from the smallest number of edges—the so called *shortest* path—but in some case any path could fit—this is the case, e.g., in the interdomain routing. The mechanism for creating a route can work on a flow basis—we speak about *connection oriented* networks—or on a datagram basis—the case of IP network. In the later case stability of the route is becoming additional important parameter that could influence the behavior of the whole flow (e.g., there is no reordering of datagrams within a flow in the connection oriented networks). Each path has one sending, one receiving, and zero or several *internal* nodes that are responsible for forwarding data.

However, as the networks were exposed to larger number of more sophisticated applications, more complex communicating patters emerged. The first one is a multicast, with still one sender but multitude of receivers. A simple extension is a communicating mesh, where every member of such a communicating group (the multicast group) could become a sender. Yet more complex communication patterns are seen in the *peer to peer* networks, where we may have multiple partially overlaid multicast groups communicating in parallel, it may use flooding, different cases of wave communication patterns, etc.

All the more complex communication patterns can still be expressed in our simple graph model using the sender–receiver paradigm. Multicast can be modeled by a set of sender to receiver$_i$ flows, but to express it correctly some kind of *coordination* (*synchronicity*) must be added to the model (data delivery to all receivers is expected to happen at the same time). Also, even in networks with unlimited bandwidth the simultaneous sending of all streams by just one element stress it above the optimal level (reducing efficiency of the communication scenario).

To deal with such complex communicating patterns more effectively, we have to extend our routing algorithm to find not paths, but whole subgraphs of the original graph. Flows going through such subgraph are more efficient than collection of individual send–receiver flows. The subgraphs represent *overlay networks*, that are specialized to transfer the particular flow pattern in the most efficient way.

When mapped back to the underlying network, the subgraphs extend the requirements on the internal path nodes. Simple forwarding (taking data from one link and sending them to another) is no longer sufficient, data must be duplicated and further processed to fit the communicating subgraph (overlay network) requirements. At the theoretical level this is just a simple extension, but propagating it back to the network proved to be very difficult, if not impossible work.

As an example, let's briefly discuss the IP multicast. It has been established as a family of protocols at the beginning of 80s in the last century Cheriton & Deering (1985); Deering & Cheriton (1990). IP multicast is based on a family of multicast routing protocols (how to create the appropriate subgraph of the network) and its implementation requires support at each network element both for routing and also for multicast forwarding. The IP multicast includes nodes that do datagram duplication—they must be able to forward incoming data to two or even more output links. IP multicast does not guarantee delivery of datagrams, does not provide any feedback to sender, it is in fact very simple extended forwarding scheme. All vendors of routers officially support multicast, yet it is not available on large parts of the Internet and

the situation is not expected to change in the future. Although simple, multicast still can interfere with the basic sender–receiver communication patterns, imposes more load on routers (duplication is more complicated than simple forwarding) and the multicast routing protocols can introduce instability into the basic routing. As the result, multicast may not work properly or could be switched off by network administrators if they suspect it to be the cause of a problem they have with the network[1] Diot et al. (2000); Dressler (2003a;b); El-Sayed et al. (2003).

If the IP multicast situation is far from satisfactory, what we can anticipate with more complex extensions, where data have to be not only transmitted but also processed during transmission?

We must change the paradigm—instead of expecting the underlying network to provide all the advanced functionality and increasing complexity above sustainable levels, more *isolation* and *independent deployment* of support for complex communication (and data processing) patterns is the possible answer. The isolation is provided by the overlay networks, that take care of all the new functionality by themselves. The independence of deployment is achieved through the *user empowered* approach. The overlay networks are constructed and managed (often just temporarily) by their own users, without any need for specific support from network and its administrators.

Several years ago we started to build a network environment based on the user-empowered approach for transport and processing data in IP networks. We used the concept of active networks and designed and developed an *Active Element*—a programmable network node designed for synchronous data distribution and processing, configurable without administrator's right—and used it as the basic building block for construction of complex communication patterns.

The initial phase of our research was influenced by the network-centric view. We designed an active router Hladká & Salvet (2001a), an extension of the classical router that allows users to define their own processing over individual data streams. The active network paradigm which introduced the active network elements, opened also the door to more user oriented approach. The active routers (and similar active network elements) are expected to be setup and operated by system administrators, with users "only" injecting smaller or larger programs to process their data within the network. Although the concept of active networks has been proved to provide the new functionality necessary to fulfill new requirements of data transmission and processing within the network, the whole idea collided with the conservative approach of network vendors and administrators. As the multicast experience demonstrated, it is very difficult to introduce new properties as they can interact in unpredictable way with the simpler, previously introduced protocols. Also, security concerns could not be overemphasized. A network programmable by end users is ripe for being taken completely by a hacker; this risk seen too high to be outweighted by the potential of new features.

At the same time as the active networks were developed, another paradigm that proved the value in giving control to end users emerged—the peer to peer networks. They completely

[1] To further illustrate this problem, we have performed a quick survey of Internet2 Bigvideo group mailing list archive (`https://mail.internet2.edu/wws/arc/bigvideo/`). This list was in operation from May 2003 to May 2006. It focused on education and problem solving for users of high-end video technologies in advanced academical networks like the one operated by Internet2. The list was not limited to Internet2 community and there was a significant international contribution. As a majority of the advanced video tools use multicast, 212 of total 625 messages, i.e., 34% was spent on multicast testing and debugging.

abandon the network-centric view, implementing in fact many already available network protocols once again, providing complete orthogonality (and independence) on the underlying network. The peer to peer networks are classical *overlay networks*, taking as granted only limited number of very simple properties of the underlying network and providing all the higher level functionality—searching, routing, etc.—by themselves.

However, the complete independence on the underlying network leads to inefficiency. The classical peer to peer networks could place their nodes only on the periphery of the network, where the users' stations are connected. The data distribution pattern required by the content (which the peer to peer network understood) may fit very poorly into the actual underlying network topology, overloading some lines while leaving other unused. Also, reliability of the peer to peer network is usually based on an overwhelming redundancy, when the same data is distributed, processed, and stored by many nodes—again a clear contradiction to the network-centric approach where the efficiency (the cost of the infrastructure) is one of the ruling paradigms.

We can see that the network-centric approach is highly efficient, but very slow in adopting new features and rather unfriendly to users. On the other hand, pure user-centric overlay approach, as represented by basic peer to peer networks is very inefficient (consuming more resources than needed in the optimal case), but it is able to introduce new features fast and can provide exactly the services the users are looking for. Another reason for that huge success is also their single purpose—the peer to peer networks are not trying to solve all the users' requirements, they focus on one service or just a small set of similar interconnected services.

Is it possible to take the positive from both approaches and leave out their negatives? Several years ago we decided to try this combination, moving from the network-centric to the user-centric approach, but not abandoning the network orientation completely. We extended the active router model to fit into the user-centric paradigm. The original active router and its implementation was based on Unix operating system and exploited both the kernel and user components. Its installation and deployment thus required system administrator's privileges that ordinary user may not have. As the next step, we completely redesigned the active router to become Active Element (AE), working in the user space of any operating system only. We obtained a fully user controlled element, that can be installed on any machine user has access to, without any specific privileges (e.g., on a server that is more strategically placed within the network than end user desktop machine). However, the AE design still followed basic network-centric pattern, being an evolutionary successor of active router, and thus became a keystone for the distribution and processing infrastructure, not a node in a peer to peer network. We still differentiate between an infrastructure and clients, but we put both into users' hands.

The user controlled Active Element is a very strong and flexible component to build different distribution schemes. We started with an infrastructure for virtual multicast. We used this infrastructure to study properties of the serial communication schema for group synchronous communication instead of the parallel communication model of the native multicast. While we had clearly demonstrated its advantages, especially in the area of security and reliability, the limited scalability remained the major disadvantage and it became our natural next research target. Instead of using just a single AE to do all the processing and distribution, we designed a network of AEs with distinct control and data planes. This separation allowed us to use the peer to peer principles at the control plane, taking advantage of the properties of peer to peer networks like robustness and very high scalability. The inherent low efficiency of peer to peer networks does not play significant role, as the amount of control data is al-

ways limited. The result is an easily configurable and fault tolerant network of AEs with a reasonably high throughput capabilities.
However, the scalability is not one dimensional issue. While the network of AEs addressed the scalability in terms of number of clients supported, very high quality video (e.g., that used in the cinema theaters) generates so huge amount of data that may not be processed by a single AE. Therefore, we extended our work on scalability to increase the AE processing capacity through their internal parallelization. The parallelized AE runs on a cluster with fast internal interconnect and is capable of processing in near real time even 10 Gbps data stream.
All this research and development would not be complete without an actual deployment. Putting the AEs and their networks into production use provided a very valuable continuous feedback on their design while experimentally testing their properties. The AEs were used to build an infrastructure for collaborative environment used by several geographically distributed groups of researchers. Requirements from these groups initiated further research into support of advanced communication and collaboration features like moderating or subgrouping. The AEs started to play a role of directly controlled user tool to support these advanced properties. This confirmed the strength of the general concept of user empowered building blocks for data processing and distribution networks.
In another environment we used the idea of overlay network with AEs capable to provide new functionality for the stereoscopic video streams synchronization. A simple software implementation running on commodity hardware is able to synchronize two streams of stereoscopic digital video (DV, 25 Mbps) format successfully even when the original streams are highly desynchronized. The penalty of the synchronization is increased latency, as the "faster" stream must wait for data in the slower stream, plus some processing latency is added to the final perceived delay. While this delay may be problematic in interactive implementation, we demonstrated that the AE-based synchronization element can be easily used for synchronized unidirectional stereoscopic streaming to multiple end users even in highly adverse and desynchronizing network conditions Hladká et al. (2005). While the stereoscopic streaming may not be too common, this concept is usable for synchronization of stereo or multichannel (e.g., 5.1) audio streams or for synchronization of separately sent audio and video streams.
The real strength of the AEs and the whole concept of controllable overlay networks is demonstrated in the multi-point High Definition (HD) video distribution. If the HD video is to be used for a synchronous collaborative environment, uncompressed streams must be sent over the network. However, the required throughput of 1.5 Gbps per each stream was too high to be sent reliably over a native multicast in heterogeneous network over multiple administrative domains (even if it was available). The optimized AEs are able to replicate even such high demanding streams in near real time and were used to build infrastructure that supported one of the world first multipoint videoconferences using uncompressed HD video Holub et al. (2006). Later, improved AEs grouped into a network became key infrastructure for a virtual classroom that ran full semester and connected 6 sites on two continents Matyska, Hladká & Holub (2007). The Active Element network processed up to 18 Gbps bi-directional bandwidth, fully confirming the usability of the AE design.
In this chapter, we present several classes of solutions following our long term research in this field. The simplest solution to user-empowered data distribution and processing is a central Active Element (AE) described briefly in Section 2, which is a programmable modular active element, that can be run in the network easily without requiring any administrative privileges. The AE distributes and optionally also processes the incoming data, which allows for unique per-user processing capabilities—something that is impossible to do with traditional

data distribution systems like multicast. As any centralized solution, it has its positives and shortcomings: while it is easy to setup and deploy, it has limited robustness and scalability, both with respect to number of streams and the bandwidth of a single stream. When more clients are collaborating or when higher robustness is needed, the AEs may be deployed as static or dynamic self-organizing AE networks shown in Section 2.1. This field has been studied thoroughly from the data distribution efficiency and robustness point of view by many groups previously and the most relevant body of work is referenced in Section 2.1. Our view here is, however, more general, focusing not only on mere multicast-like data distribution, but also on the possibilities enabled by additional data processing, operation in adverse networking environments, self-organization, etc. Another step forward needs to be taken when bandwidth of a single stream exceeds capacity of any single AE in the AE network. Utilizing properties of real-time multimedia applications and data distribution protocols, we have designed a distributed AE described in Section 2.2 that can be deployed on tightly coupled clusters—but this solution becomes very complex when not only the data distribution but also data processing is required. We demonstrate applications which have been built on top of these technologies for synchronous data distribution and processing in Section 3. Related work is summarized in Section 4 and we conclude with some remarks on directions for future research in Section 5.

2. Active Elements

The Active Element (AE) Hladká et al. (2004) is a programmable element designed for synchronous data distribution and processing while minimizing the latency of the distribution. The word "reflector" is also being used in this context, which only refers to data distribution capabilities. Since our approach is far more general and close to idea of active networks, we have resorted to using the Active Element name. The architecture of the AE is flexible enough to allow implementation of required features while leaving space for easy extensions. If the data is sent to all the listening clients and all the clients are also actively sending, which is a standard scenario for collaborative group of participants, the number of data copies is equal to the number of the clients, and the limiting outbound traffic grows with $n(n-1)$, where n is the number of sending clients.

From general point of view the AE is a user-controlled modular programmable router working on the application layer. It runs entirely in user-space of the underlying operating system and thus it works without the need for administrative privileges on the host computer. AEs are based on our active router concept described in Hladká & Salvet (2001b), building on the same principles of modularity, but adding the user-empowered approach. The AE architecture is is shown in the Figure 1.

Data processing architecture.

Data routing and processing part of the AE comprises *network listeners*, *shared memory*, *a packet classifier*, *a processor scheduler*, number of *processors*, and *a packet scheduler/sender*.

The network listeners are bound to one UDP port each. When a packet arrives to the listener it places the packet into the shared memory and adds reference to a *to-be-processed queue*. The packet classifier then reads the packets from that queue and determines a path of the data through the processor modules. It also checks with routing AAA module whether the packet is allowed or not (in the later case it simply drops that packet and creates event that may be logged). Zero-copy processing is used in all simple processors (packet filters), minimizing processing overhead (and thus packet delay). E. g. for simple data multiplication, the data

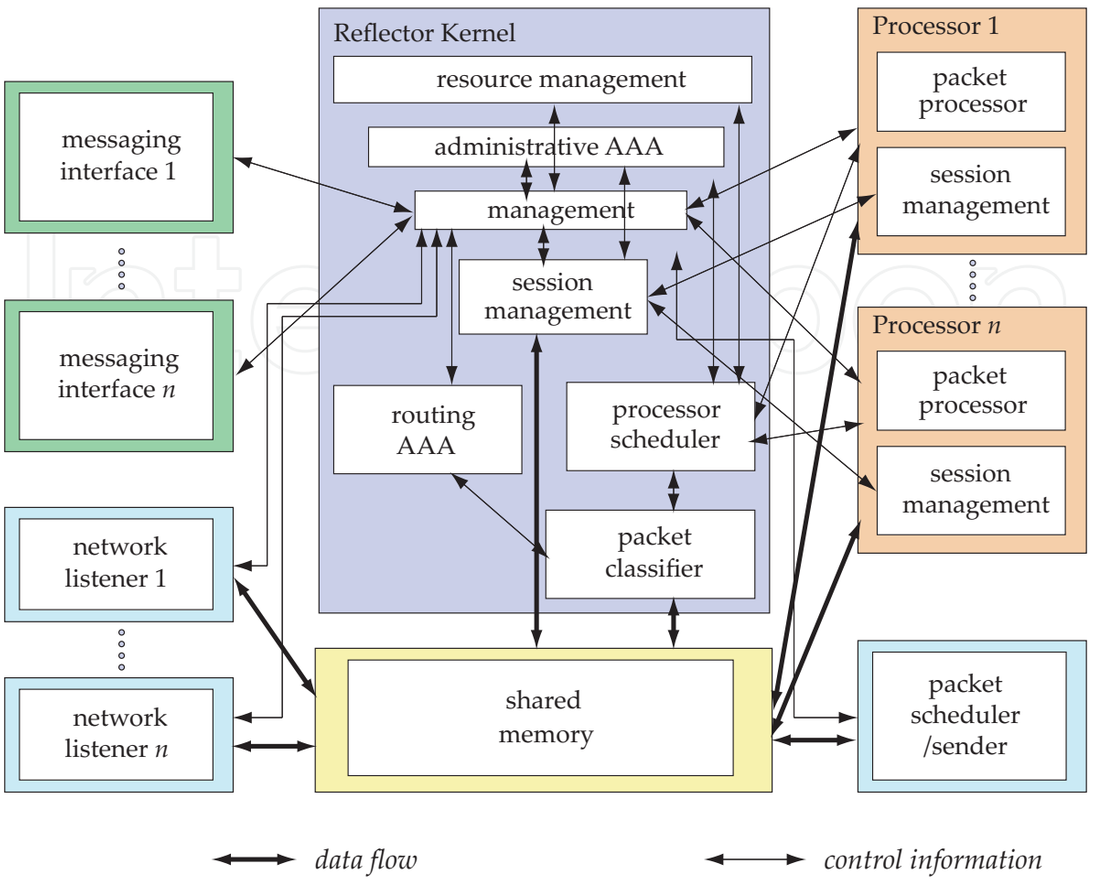


Fig. 1. Architecture of Active Element with its individual modules and interactions.

are only referenced multiple times in the packet scheduler/sender queue before they are actually being sent. Only the more complex modules may require processing that is impossible without use of packet copies.

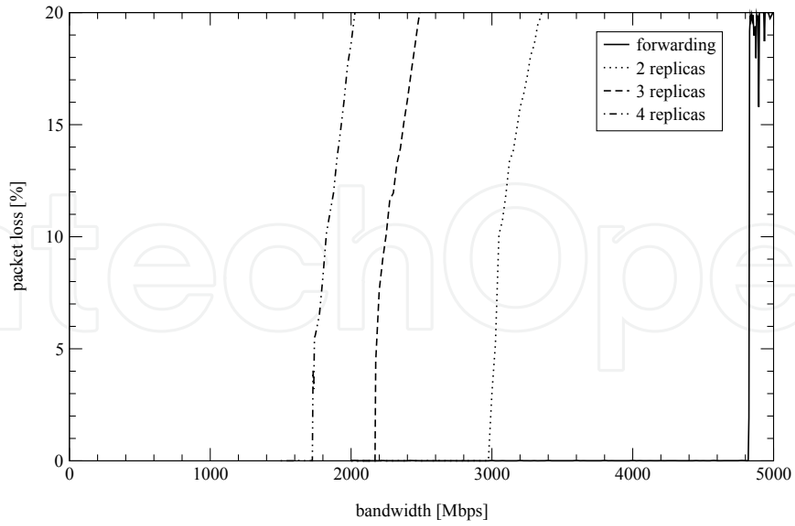
The *session management* module follows the processors and fills the distribution list of the target addresses. The filling step can be omitted if data passed through a special processor that filled the distribution list structure and marked data attribute appropriately (this allows client-specific processing). Processor can also poll session management module to obtain up to date list of clients for specified session. Session management module also takes care of adding new clients to the session as well as removing inactive (stale) ones. There are two ways of adding clients for a session at the AE: implicit mode and explicit mode. In the implicit mode, when new client sends packets for the first time, session management module adds client to the distribution list (data from forbidden client has already been dropped by packet classifier). This mechanism is designed to work with the multimedia systems like MBone Tools suite. The explicit mode requires some specific action to be taken by the user or application to register for the session at the AE, be it RTSP protocol Schulzrinne et al. (1998) or direct interaction through one of native messaging interfaces of the AE. Information about the last activity of a client is also maintained by the session module and is used for pruning stale clients periodically in the

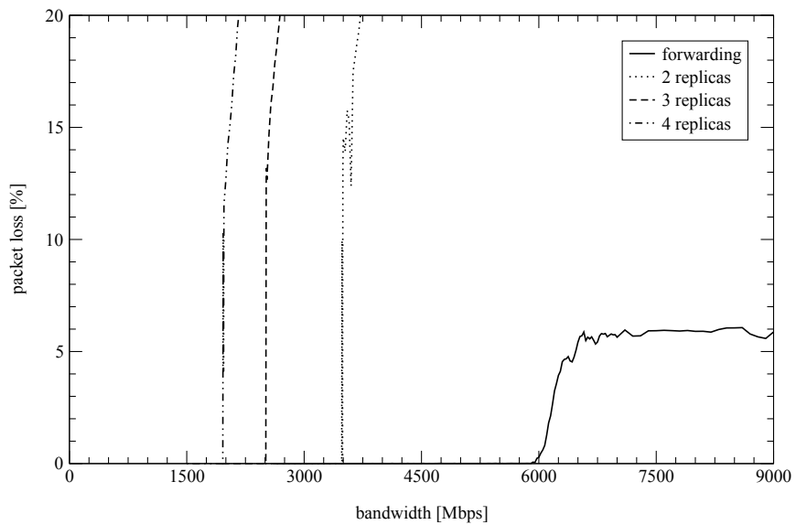
implicit mode. Even when distribution list is not filled by the session management module, packets must pass through it to allow addition of new clients and removal of stale ones.
When the packet targets are determined by the router processor a reference to the packet is put into the *to-be-sent queue*. Then the packet scheduler/sender picks up packets from that queue, schedules them for transmission, and finally sends them to the network. Per client packet scheduling can also be used for e. g. client specific traffic shaping.
The *processor scheduler* is not only responsible for the processors scheduling but it also takes care of start-up and (possibly forced) shutdown of processors which can be controlled via administrative interface of the AE. It checks resource limits with routing AAA module while scheduling and provides back some statistics for accounting purposes.
Architecture of management.
Communication with the AE from the administrative point of view is provided using *messaging interfaces*, *management module*, and *administrative AAA module* of the AE. Commands for the management module are written in a specific *message language*.
The administrative part of the AE can be accessed via secure messaging channels such as HTTP with SSL/TLS encrypted transport or SOAP with GSI support. The user can authenticate using various authentication procedures, e. g., combination of login and password, Kerberos ticket, or X.509 certificate. Authorization uses access control lists (ACLs) and is performed on a per-command basis. Authentication, authorization, and accounting for the administrative section of the AE is provided by an administrative AAA module. Each of these interfaces unwraps the message if necessary and passes it to the management module. A message language for communication with the management module is called Reflector Administration Protocol (RAP) described in Denemark et al. (2003).
Prototype implementation and performance evaluation.
In order to evaluate the behavior of AE on recent high-performance infrastructure, we have set up a testbed comprising sender and receiver machines (each 2× AMD Opteron 2.4 GHz, 2 GB memory, Linux 2.6.9 SMP kernel) and a machine running the AE (2× dual core Intel Xeon 3.0 GHz, 8 GB memory, Linux 2.6.19 SMP kernel). The sender machine was equipped with Chelsio T110 and both the receiver and the AE machine with Myricom Myri-10GE NICs. All the three machines were connected to a 10GE Cisco 6506 switch.
The performance was measured using two implementations of the AE: the full featured complex version described above[2] (denoted as AE) and a high-performance simplified version including only one receiving and one sending thread, which was designed for HD video distribution Holub et al. (2006) (thus denoted as HD-AE). The performance is summarized in Fig. 2. The results indicate that even the more complex version is capable of distributing the uncompressed HD video for up to 4 participants when Jumbo frames are used, which is necessary for this application anyway.

2.1 AE Networks

As the scalability of AE is limited especially with respect to the number of data streams (clients), the concept of single AE has been extended to a network of AE Holub et al. (2005) while preserving its processing capabilities through modularity and retaining the user-empowered approach to maximum extent. Its architecture features separated data distribution plane and control plane: while the data distribution is optimized for maximum performance and minimum latency, the control plane has to provide maximum robustness even at

[2] The AE concept has been implemented in C language for Unix-like operating systems under code name RUM2. `http://miro.cesnet.cz/software/software.cz.html`.





replica(s)	AE		HD-AE	
	1500 B	8500 B	1500 B	8500 B
1	1450	4820	2700	6100
2	1000	2975	1660	3490
3	775	2170	1140	2510
4	600	1725	825	1960

Fig. 2. Performance of the modular full-featured AE compared to highly simplified version optimized for HD video distribution (HD-AE). Stabilization in the upper right graph is because of sender card saturation above 6.5 Gbps. The table below the graphs shows maximum stream bandwidth [Mbps] distributed with less than 0.1 % packet loss for both standard and Jumbo frame sizes.

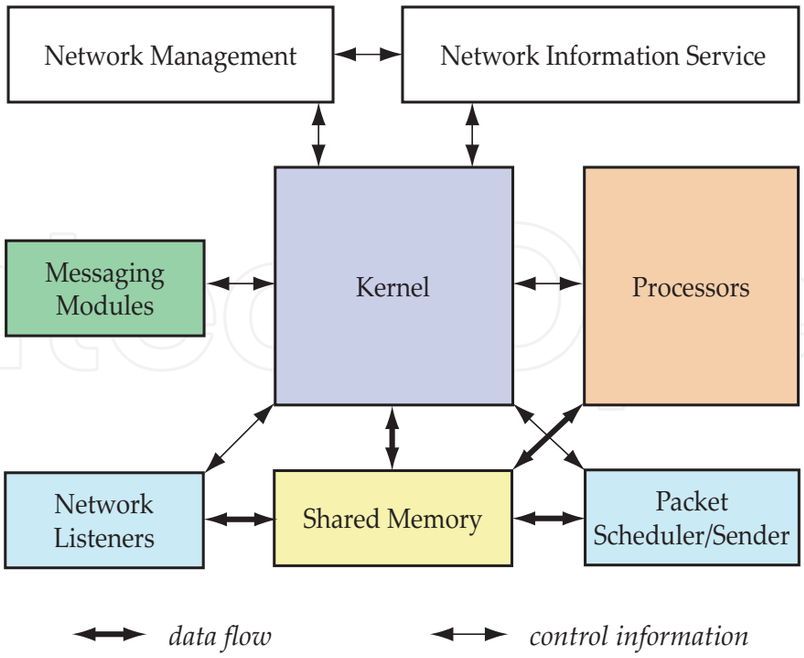


Fig. 3. Architecture of Active Element with Network Management and Network Information Service modules.

cost of performance. The control plane is responsible for management actions of the AE network like monitoring, reconstruction of the network after node or network link failure and has to survive all the perturbations. Thus we have chosen a P2P architecture of the control plane which exhibits very strong resilience. The data plane on the other hand may be dynamically rebuilt based on the information from the control plane; even the data distribution model may change.

The AEs has networking capability, i.e., inter-element communication. The network management is implemented via two modules dynamically linked to the AE: Network Management (NM) and Network Information Service (NIS) as shown in Fig. 3. The NM takes care of building and managing the network of AEs, joining new content groups and leaving old ones, and reorganizing the network in case of link failure. The NIS gathers and publishes information about the specific AE (e.g., available network and processing capacity), about the network of AEs, about properties important for synchronous multimedia distribution (e.g., pairwise one-way delay, RTT, estimated link capacity), and also information on content and available formats distributed by the network.

The data distribution plane is designed using loadable plug-ins to enable incorporating various distribution models. A number of suitable models has been proposed previously by many independent groups in the past, most of which fall into one of the two categories: (1) mesh first distribution models like Narada Chu et al. (2000), Delaunay triangulation Liebeherr & Nahas (2001), Bayeux Zhuang et al. (2001), and (2) tree first models like YOID Francis (2000), TBCP Mathy et al. (2001), HMTP Zhang et al. (2002), SHDC Mathy et al. (2002), NICE Banerjee et al. (2002), Overcast Jannotti et al. (2000), ZIGZAG Tran et al. (2003). Some other models may also be found in El-Sayed et al. (2003); Li & Shin (2002).

Given the data processing capabilities of the AE, the usefulness of AE networks goes beyond pure data distribution models. AEs in the network can be specialized in performing various transformation of the data based on user request (e.g., AE running on host with enough CPU power and sufficient network capacity can perform transformation of the data from high-bitrate to low-bitrate). However, combinations of data distribution and data processing makes makes scheduling problem particularly hard and first approaches have only been studied recently using self-organizing CoUniverse platform Liška & Holub (2009).
Prototype implementation of the AE networks with P2P control plane based on JXTA-C[3] has been demonstrated in Procházka et al. (2005). A few simple optimizations to default JXTA settings improved the performance significantly for synchronous applications with a limited number of participants where down-time minimization is required despite increasing communication overhead, thus making it suitable control-plane middleware.
The AE network is also designed to facilitate communication in adverse networking environments, i.e., environments where the network communication is obstructed by firewalls, network address translators (NATs) and proxy servers. The data may be tunneled over TCP instead of usual UDP, it may even mimic using HTTP and tunnel the data over HTTP proxy. The AE may also be augmented by employing a VPN Holub et al. (2007) such as OpenVPN[4], which boosts pervasivity, as it allows even tunneling through HTTP and SOCKS proxy servers. VPN also enables deployment of strong authentication and very secure data encryption protocols. Similar approaches have also been described in Alchaal et al. (2002). The solution that integrates these features directly into AE modules Bouček (2002); Salvet (2001) has significant advantages despite having a more demanding implementation: it allows for dynamic failure recovery properties in case of AE node failure or network link failure, as the client may join the AE network using another AE node that is still available and reachable.

2.2 Distributed Active Element

Another scalability issue regarding both single AE and AE networks is scalability with respect to the bandwidth of each individual data stream. In oder to improve on this, we have designed a distributed AE Holub (2005); Holub & Hladká (2006), intended to be run on tightly coupled clusters with low latency network interconnection for the control plane and high-bandwidth interfaces for the data plane. The distributed AE splits a single stream into multiple substreams, which are processed in parallel—thus possibly introducing packet reordering. This is significantly different from general purpose load distribution systems like LACP IEEE 802.3ad protocol, which have to avoid the packet reordering and therefore a single data stream is processed sequentially[5]. The distributed AE includes distribution unit to distribute the data to the parallel processing units, and aggregation unit, which aggregates the data from the parallel units.
Limited synchronization and FCT. The basic idea behind distributed AE utilizes the fact, that most of the synchronous multimedia applications use non-guaranteed data transport like UDP and thus they need to adapt to some packet reordering. However, significant data reordering may either not be adapted upon or it results in latency increase as substantial buffer-

[3] `http://www.jxta.org/`
[4] `http://openvpn.net/`
[5] This is done by using data flow identifiers hash to assign each data flow to a specific link of the of the aggregated link group. Thus each single data flow must not exceed capacity of the single link.

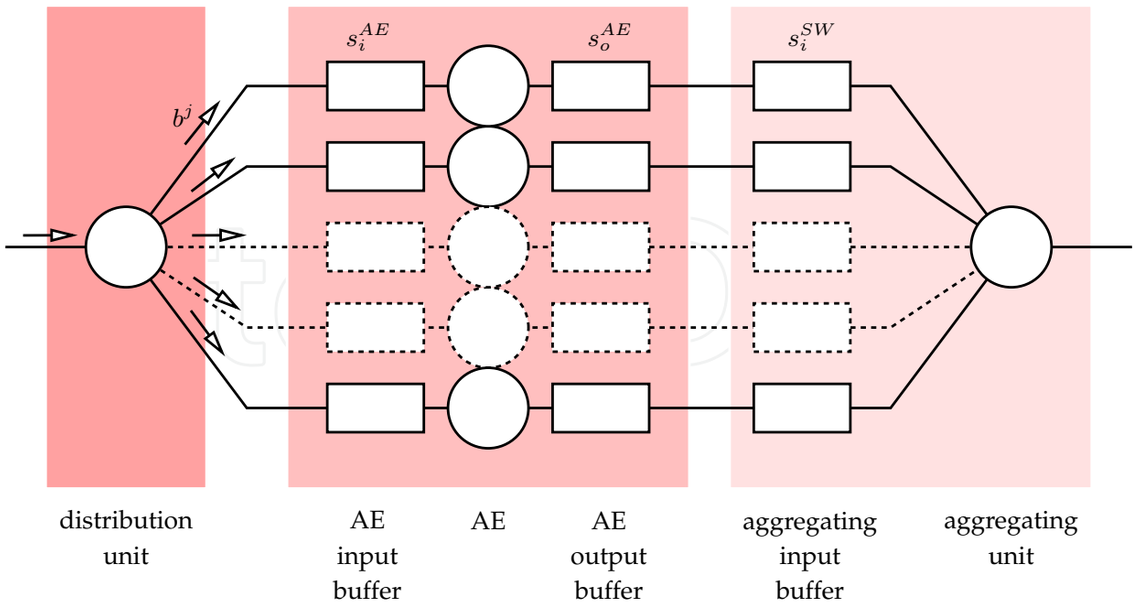


Fig. 4. Model of the ideal distributed AE with ideal aggregation unit.

ing is necessary to reorder the packets. Without any explicit synchronization, the maximum packet reordering can be

$$n(s_i^{AE} + s_o^{AE} + s_i^{SW} + 1)$$

where n is number of parallel paths and s_i^{AE}, s_o^{AE}, s_i^{SW} are buffer sizes on input of the AE, output of AE, and input of the aggregation unit respectively, as shown in Figure 4.
In order to decrease packet reordering introduced by the distributed AE, we have introduced a distributed algorithm for achieving less packet reordering compared to no explicit synchronization. The nodes are ordered in a ring with one node elected as a master node and they circulate a token which serves as a barrier so that no node can run too much ahead with sending data. After reception of the token containing the current "active" round number, each non-master node passes on the token immediately and may send only the data from the round marked in the token until it receives to token again. When the master node receives the token from the last node in the ring, it finishes sending the current round, increments the round number in the token a passes on the token. The mechanism is called *Fast Circulating Token* (FCT) since the token is not held for the entire time period of data sending as usual in the token ring networks.
Because of real world implementation of data packet sending in common operating systems, we assume that sending procedure for a single packet is non-preemptive. Further we assume that token reception event processing has precedence over any other event processing in the distributed AE. However, as the data sending is non-preemptive, if the token arrives in the middle of data packet sending, it will be handled just after that packet sending is finished.
After more detailed analysis Holub (2005), it can be shown the maximum reordering induced by an ideal distributed AE with FCT egress synchronization and ideal aggregating unit is

$$n(s_i^{SW} + 3),$$

when all queues operate in FIFO tail-drop mode. Thus the receiving application can adapt its buffer size to this upper bound. On custom hardware, the FCT protocol can be adapted to provide no packet reordering at all (called Exact Order Sending. More in-depth analysis can be found in Holub (2005); Holub & Hladká (2006).

Prototype implementation and experimental evaluation. Prototype implementation of the distributed AE is implemented in ANSI C language for portability and performance reasons. The implementation comprises two parts: a load distribution library and the distributed AE itself.

Because of lack of flexible enough load distribution hardware unit, we have implemented it as a library, which allows simple replacement of standard UDP related sending functions in existing applications and allows developers to have defined type of load distribution—either pure round robin or load balancing.

Each parallel AE uses threaded modular implementation based on architecture described above. Internal buffering capacity of each AE node has been set to 500 packets. Explicit synchronization using FCT protocol has been implemented using MPICH implementation[6] of MPI built with low-latency Myrinet GM 2.0 API[7] (so called MPICH–GM). Prototype implementation has been tested on Linux.

For cost-effective prototype implementation, the aggregation unit was a implemented as commodity switch with sufficient capacity of internal switching fabrics.

The experimental results obtained on 10GE infrastructure, revealing that the distributed AE is capable of completely saturating sender machine in a testbed similar to the one used for AE performance evaluation above. Up to 8 parallel units were used for the measurement, connected using Gigabit Ethernet NICs into GE ports of the Cisco 6506. Myrinet-2000 NICs and switch were used for the low-latency interconnection. Packet distribution was implemented as user-land UDP library and the aggregation was performed by the Cisco 6506 switch. When the FCT protocol is used, the experimental evaluation showed the maximum packet reordering is below 15 for 8 parallel units, which makes it comparable to long-haul networks of good quality. Without the FCT, the maximum reordering was up to 111 for the same setup, i.e., one magnitude worse. Typical results can be seen in Figure 5.

The distributed AEs can also be incorporated into an AE network using the same approach described in the previous chapter. However, because of running on more complicated infrastructure, the setup and start is more complex than for a single AE and thus the system has worse fail-over behavior compared to the network of simple AEs. Another complication of the distributed AE is in the processing of the passing data, which requires development of parallel programming paradigms similar to MIT StreamIt Thies et al. (2002). The processing may follow one of two possible approaches: (1) a context is maintained within one parallel unit only (requires either that all the data requiring the same context to be processed in are processed with one parallel unit only, or per-packet processing without a context is used), or (2) the context is maintained within a subset of parallel nodes using the low-latency interconnection of the cluster. These approaches will be further investigated in the future.

[6] `http://www-unix.mcs.anl.gov/mpi/mpich/`
[7] `http://www.myri.com/scs/GM-2/doc/html/`

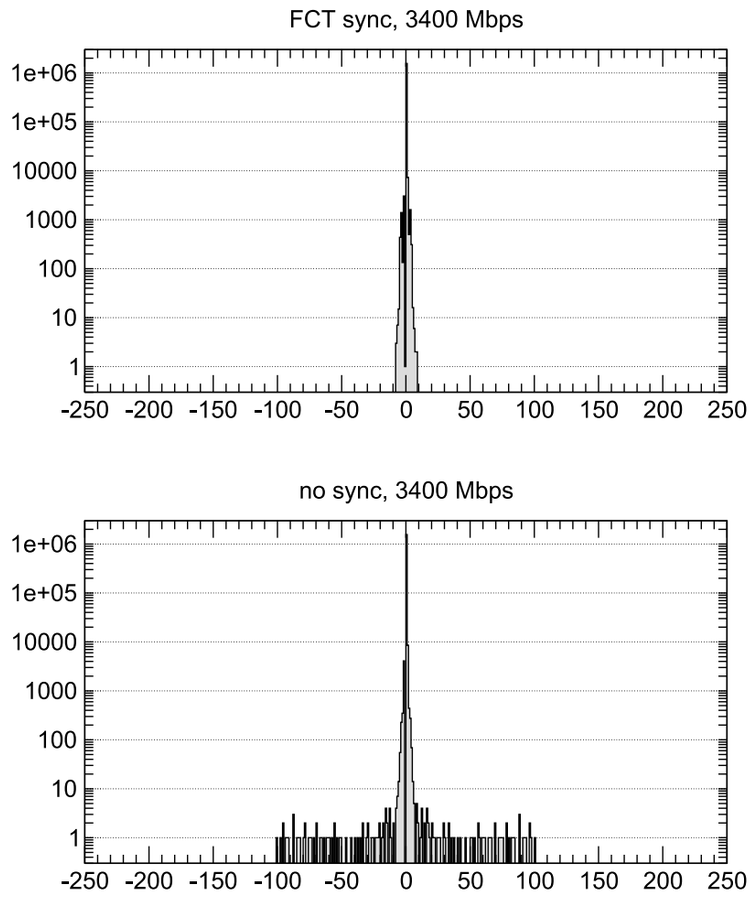


Fig. 5. Packet reordering distribution with FCT and without synchronization, for 8 parallel units and 3.4 Gbps per data flow.

3. Applications

Applications of virtual multicast range from simple user-empowered data distribution to complex data parallel data processing tasks and per-user data processing. Overlay network creating virtual multicast can be also used to distribute data strongly protected environments. These use cases are further discussed in this chapter.

3.1 Data Distribution

The AEs have been used routinely by different groups for collaboration, mostly with MBone Tools[8], DVTS[9], and uncompressed HD video based on UltraGrid Holub et al. (2006). A recent demonstration of uncompressed HD video with bandwidth usage of 1.5 Gbps per data stream at SuperComputing | 06 conference[10] used a network with 3 optimized HD AEs in StarLight (Chicago, USA) and achieved sustained aggregated data rate of 18 Gbps without any packet

[8] http://www-mice.cs.ucl.ac.uk/multimedia/software/
[9] http://www.sfc.wide.ad.jp/DVTS/
[10] https://sitola.fi.muni.cz/igrid/index.php/SuperComputing_2006

loss. As an alternative setup, we have also used a combination of an AE with multiplication on optical layer (optical multicast), which is, however, far from user-empowered as it requires both direct access to Layer 1 network and installation of specialized hardware directly into the network. The high-performance static AE network has also been used in production for uncompressed HD video distribution for a distributed class on high-performance computing taught by prof. Sterling at Louisiana State University Matyska, Holub & Hladká (2007). In this case, dedicated λ-circuits spanning 5 institutions across the USA and one in the Czech Republic were used and, therefore, a static configuration was the most appropriate as the circuit topology was also statically configured. The 1.5 Gbps streams were distributed up to 7 locations as shown in Figure 6.

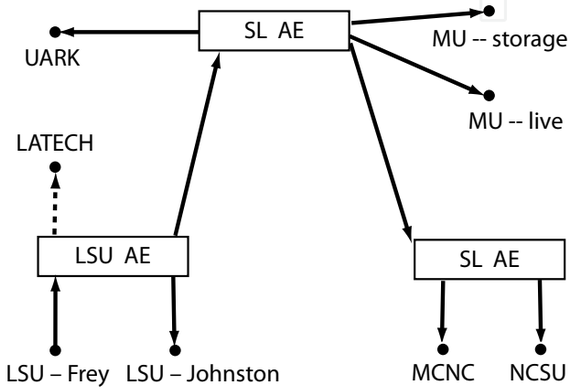


Fig. 6. Data distribution for LSU HPC Class by prof. Sterling based on uncompressed HD video with 1.5 Gbps per stream.

With much lower bandwidth per stream but many more clients served, another AE network is also used for streaming data distribution using VLC at the Masaryk University to get the live video feeds from the lecturing halls even over networks without multicast support. Furthermore, it is used for tunneling the data to the student dormitories which have very adverse networking environments. This AE network also supports transcoding as described below.

3.2 Stream transcoding

Typical application of processing on an AE is stream transcoding. For live video stream distribution from several lecturing halls at the Masaryk University, a transcoding processor for the video and audio streams has been implemented as an AE module Liška & Denemark (2006). It uses VideoLAN Client[11] (VLC) as the actual transcoding back-end, thus giving us a large variety of supported formats for both input and output. The transcoding module communicates with VLC in three ways: the source data is delivered using Unix standard I/O, the transcoded data is received from VLC using a local UDP socket in order to receive the data appropriately packetized, while a local telnet interface is used for remote control of VLC.
For the specific application, the distribution schema is shown in Fig. 7. There are basically two types of video stream sources: an MPEG-2 hardware encoder such as Teracue ENC-100 or a regular MPEG-4 streaming PC with video capture card and VideoLAN Client installed.

[11] `http://www.videolan.org`

In both cases the video stream is generated as a standard MPEG Transport Stream (MPEG-TS) at 2 Mbps and sent using unicast to the AE for further processing. The original data is available to the students either using unicast (AE) or multicast from the AE, or they can watch transcoded video from the gateway AE. Students then use VLC again for rendering the streams at their computers. This allows us to provide students at the high-speed networks with the maximum quality video, while students with slower networks (e.g., in dormitories) are also supported and may participate in the class using transcoded streams with a lower bitrate. Depending on the settings, the transcoding can consume a considerable amount of processing power and therefore the transcoding AE has significantly lower distribution capacity. As a result it is set up at the beginning of the low bandwidth link working as a gateway or bridge only, while another AE is use to actually distribute the transcoded data at large.

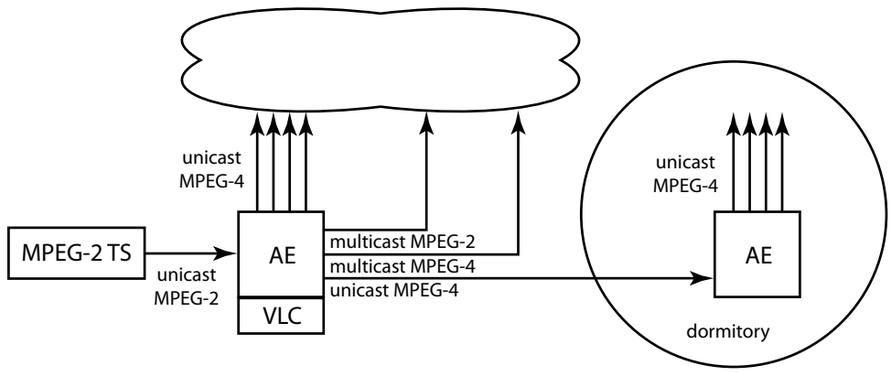


Fig. 7. Video stream distribution schema for the live streaming from lecturing halls.

Performance evaluation

In order to evaluate efficiency and scalability of this solution, we have performed a series of performance and latency measurements.

For performance measurements, we have used the following testbed: the AE was running on a computer with the dual-core Pentium D at 3 GHz, 1 GB RAM, and a Gigabit Ethernet (GE) NIC Broadcom NetXtreme BCM5721. Client computers were furnished with two Intel Xeon 3 GHz processors, 8 GB RAM, and a GE NIC BCM 5708. The testbed was interconnected using two HP Procurve GE switches (2824 and 5406zl). All the computers were running Linux kernel version 2.6. We have optimized buffer settings on NIC to 1 MB to improve the performance. For transcoding, VLC 0.8.6b with ffmpeg library using libavutil 49.4.0, libavcodec 51.40.2, and libavformat 51.11.0 was used. Source video for transcoding was in MPEG-2 format with full PAL resolution (768×576) at 6 Mbps bitrate. MPGA audio accompanied the video and both streams were encapsulated in MPEG Transport Stream format. The output stream was MPEG-4 with 576×384 resolution at 1 Mbps bitrate, audio bitrate 128 kbps, all encapsulated again in MPEG TS. Scalability and resource utilization is shown in Figure 8.

We have also performed a similar experiment for H.264 output video with full PAL resolution at 2 Mbps bitrate using x264 library[12], but this didn't work as the conversion used 100 % CPU capacity which resulted in visible packet loss in the image.

[12] `http://www.videolan.org/developers/x264.html`

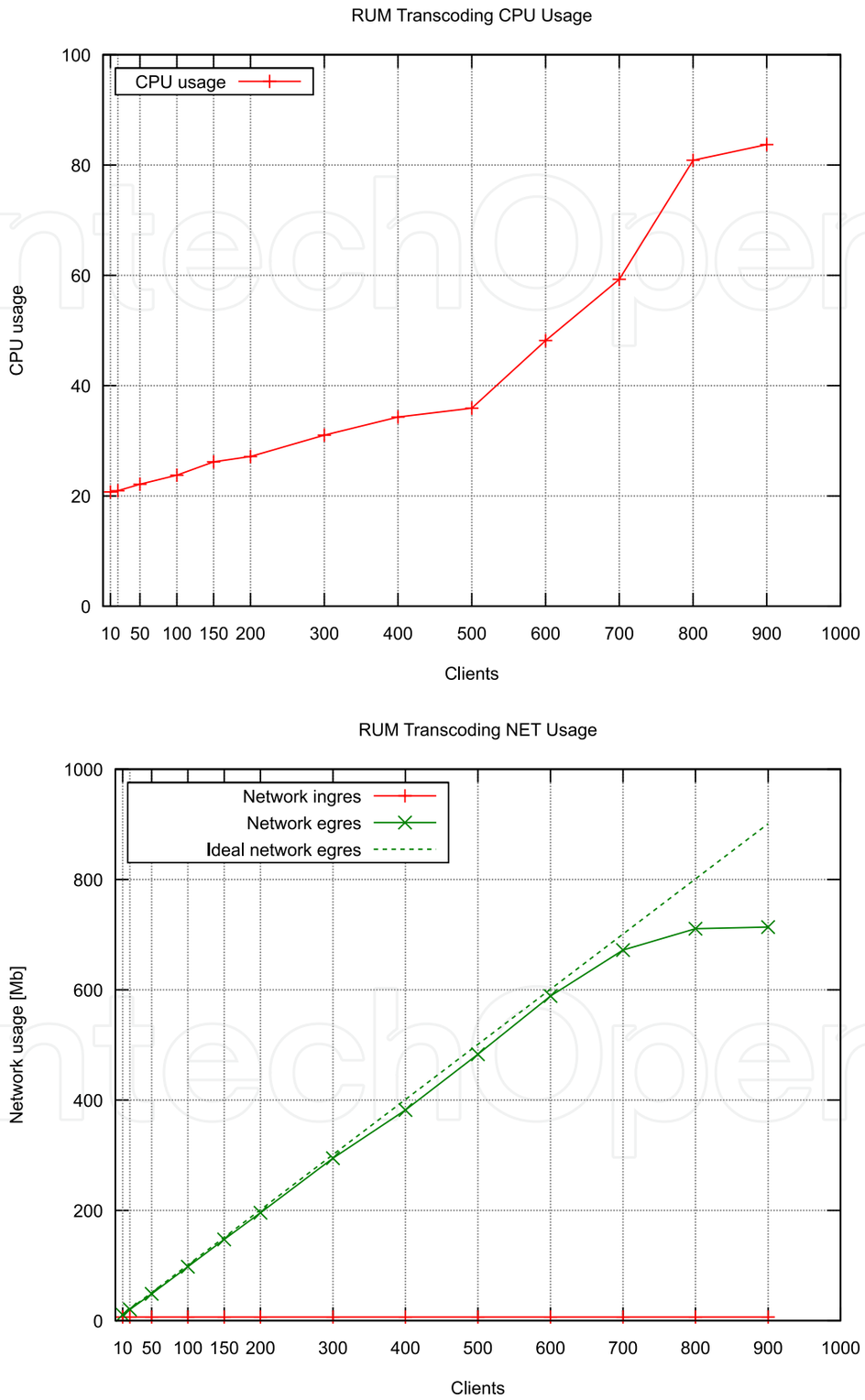


Fig. 8. Performance characteristics of transcoding AE (1 MB buffer or NIC).

Output encoding	*Measure latency [ms]*
MPEG-4	1220 ± 20
MPEG-4, keyint $= 1$	1240 ± 20
MJPEG, keyint $= 1$	1200 ± 20
H.263, keyint $= 1$, res. 704×576	1180 ± 20

Table 1. Transcoding AE latency measurements results.

Using the same setup, we have measured also the latency of the transcoding AE. The results are summarized in Table 1. Obviously, this implementation of transcoding provides too much latency for interactive video communication, but it is perfectly valid for streaming purposes described above. The latency can be decreased to tens of milliseconds when implemented directly as an AE module, not dependent on external transcoding tool—the latency added by a simple AE module that just passes on the raw data in zero copy mode is 0.238 ms on given infrastructure.

3.3 Video stream composition

Large group collaboration may easily result in too many windows at client sites (typical scenario for AccessGrid[13]), and clients may not have sufficient power or desktop space to render them all. In cases like this, it may be advantageous to down-sample the video streams and compose several of them into a single stream directly on the AE. The same technique is implemented in MCUs for H.323/SIP, but it was unavailable for MBone Tools. The first version of video compositor Holer (2003) has been adapted to fit into the modular AE architecture as a processor. This processor is based on the VIC tool McCanne & Jacobson (1995) and thus it supports exactly the same set of video formats and the result is seen in Fig. 9. Up to four video streams can be composed into one output stream. Input video formats are auto-detected, the processor is able to work with different formats simultaneously. The output video format is configurable by the end user.

Fig. 9. Example of video stream composition at AE using VIC video clients.

[13] http://www.accessgrid.org/

3.4 Operation in Adverse Networking Environments and Security

The real-time communication for healthcare purposes is unique because of the two classes of interconnected requirements: security and ability to operate even in heavily protected networking environments. The security is necessary as the specialist often need to communicate very sensitive patients data. Because of the security requirements, the healthcare institutions are usually trying to implement the most restrictive networking scenarios. E.g., we have been collaborating with a hospital that has its network protected by a firewall and hidden behind a NAT, that allows only HTTP traffic, which has to pass through two tiers of proxies. However, even the specialists from this hospital need to communicate with their colleagues. The AE approach combined with VPNs have been deployed successfully for several healthcare related projects and we were able to include even the institute mentioned above. As shown in Holub et al. (2007), the OpenVPN approach only has a minimal impact on the performance of collaborative tools. Another important feature that we are developing in this field is efficient aggregation of individual media streams—not only the video streams as discussed above—as some of the institutions, especially in developing countries, have only very limited Internet connection capacity.

4. Related work

Distribution of multimedia data over IP network leads to a multicast schema Almeroth (2000). However, as the native multicast solution is not always appropriate (e.g., for many small groups which is characteristic for interactive collaboration as it has been designed for small number of large groups), reliable, or even available, other distribution schemes were developed following the approach of multicast virtualization El-Sayed et al. (2003); Li & Shin (2002), e.g., Mtunnel Parnes et al. (1998) and UMTP Finlayson (2003). While many theoretical concepts for data distribution were developed namely during 1998 – 2003 period (see the data distribution models referenced in Section 2.1), the practical approaches are still usually based on a central distribution unit or static topologies like the H.323 MCUs or reflectors provided in the Virtual Room Videoconferencing System (VRVS)[14]. The successor of VRVS called Enabling Virtual Organizations (EVO)Galvez (2006) is based on a self-organizing system of reflectors, again not empowering the end-user with tools to change the distribution topology. High-perormance dynamic data distribution system used for distribution of 200 Mbps compressed 4K video streams designed by NTT is called Flexcast Shimizu et al. (2006). Another application-level multicast called Host Based Multicast (HBM) has been proposed in El-Sayed (2004). The HBM author also investigated a combination of an IPsec based VPN environment—while useful for data protection, it doesn't improve on adaptability of HBM for adverse networking environments. Other simpler UDP packet reflectors include rcbridge Buchhorn (2005), reflector[15], and Alkit Reflex[16]. However, all these systems are primarily focused on pure data distribution and most of them even neglect the user-empowered view, thus differing significantly from our highly modular and user-empowered AE based on active network concept.

Another relevant field of work is parallel stream-oriented processing and programming of such systems, which is of high importance for the distributed AE. A parallel programming paradigm, that might be suitable for distributed AE programming, has been proposed in MIT

[14] http://www.vrvs.org/

[15] http://www.cs.ucl.ac.uk/staff/s.bhatti/teaching/z02/reflector.html

[16] http://w2.alkit.se/reflex/

StreamIt system Thies et al. (2002). It enables efficient parallelization of the data processing based on sent data structure and processing dependencies. Its suitability and possible adaptation will be further investigated.

5. Future work

In this chapter, we have explained basic principles of multicast virtualization, presented a framework of Active Elements, designed for user-empowered synchronous data distribution and processing. Depending on target environment and the streams that are being distributed, the AEs may be deployed as a single central entity, or as a network of AEs for increased scalability with respect to number of clients and increased failure resiliency, or as a distributed AE to improve scalability with respect to the bandwidth of individual data stream. We have demonstrated a number of applications both for data distribution and processing.

In the future, there are at least several areas to focus on. Utilizing a single AE, we would like to introduce multi-level QoS approach to provide strict user and stream separation. This is especially important when an AE is used for data processing. We would like to use a virtualization-based approach to achieve this, and the virtual machines may also be used for "programming" the AE, as the user may "inject" the whole virtual machine into the AE. For the AE network, we would like to develop more complex signaling protocols to improve diagnostics (e.g., failure information needs to be distributed not only inside the AE network, but also to the influenced users in some way). The virtual machine approach will also be used to simplify migration of the processing modules in the AE network. Last but not least, we will further investigate programming paradigms suitable for the distributed AE to enable truly parallel stream processing.

This field of data distribution in overlay networks has been thoroughly examined by several research groups between 1998 – 2003; some were examining the data distribution perspective, while others were also looking at security issues. We provide a much broader view of the field extending it with active network and user-empowered approaches. We have demonstrated that while the research interest in this field dropped since 2003, new useful techniques can still be invented and there are many practical applications worth analyzing.

Larger networks of AEs that are specialized in their functionality for data distribution as well as processing goes beyond human capacity to manage such system. Thus we are researching application of self-organization principles to application orchestration Liška & Holub (2009), that could include not only AE and their networks, but also other components ranging from individual applications running at users' computers to allocation of network circuits.

Acknowledgments

This project has been kindly supported by the research intent "Optical Network of National Research and Its New Applications" (MŠM 6383917201) and "Parallel and Distributed Systems" (MŠM 0021622419). We would like to appreciate various colleagues that worked in various stages of mutlicast virtualization projects, namely Jiří Denemark, Luděk Matyska, and Tomáš Rebok.

6. References

Alchaal, L., Roca, V. & Habert, M. (2002). Offering a multicast delivery service in a programmable secure IP VPN environment., *Networked Group Communication, Fourth In-*

ternational COST264 Workshop, NGC 2002, Boston, MA, USA, October 23-25, 2002, Proceedings, ACM, pp. 5–10.

Almeroth, K. C. (2000). The evolution of multicast: From the MBone to inter-domain multicast to Internet2 deployment, *IEEE Network* **14**(1): 10–20.

Banerjee, S., Bhattacharjee, B. & Kommareddy, C. (2002). Scalable application layer multicast., *Proceedings of the ACM SIGCOMM 2002 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, August 19-23, 2002, Pittsburgh, PA, USA*, ACM, pp. 205–217.

Bouček, T. (2002). *Kryptografické zabezpečení videokonferencí (Securing videoconferences using cryptography)*, Master's thesis, Military Academy of Brno, Czech Republic.

Buchhorn, M. (2005). Designing a multi-channel-video campus delivery and archive service, *The 7th Annual SURA/ViDe Conference*, Atlanta, GA, USA.

Cheriton, D. R. & Deering, S. E. (1985). Host groups: a multicast extension for datagram internetworks, *Technical report*, Stanford University, Stanford, CA, USA.

Chu, Y.-H., Rao, S. G. & Zhang, H. (2000). A case for end system multicast, *ACM SIGMETRICS: Measurement and Modeling of Computer Systems*, pp. 1–12.

Deering, S. E. & Cheriton, D. R. (1990). Multicast routing in datagram internetworks and extended LANs, *ACM Transactions on Computer Systems* **8**: 85–110.

Denemark, J., Holub, P. & Hladká, E. (2003). RAP – Reflector Administration Protocol, *Technical Report 9/2003*, CESNET.
URL: *http://www.cesnet.cz/doc/techzpravy/*

Diot, C., Levine, B. N., Lyles, B., Kassem, H. & Balensiefen, D. (2000). Deployment issues for the IP multicast service and architecture, *IEEE Network* **14**(1): 78–88.

Dressler, F. (2003a). Availability analysis in large scale multicast networks, *15th IASTED International Conference on Parallel and Distributed Computing and Systems (PDCS2003)*, Vol. I, pp. 399–403.

Dressler, F. (2003b). *Monitoring of Multicast Networks for Time-Synchronous Communication*, Ph.d. thesis, University of Erlangen-Nuremberg.
URL: *http://www7.informatik.uni-erlangen.de/ dressler/publications/dissertation.pdf*

El-Sayed, A. (2004). *Application-Level Multicast Transmission Techniques Over The Internet*, PhD thesis, INRIA Rhône-Alpes, France.
URL: *http://www.inrialpes.fr/planete/people/elsayed/phd/phd_final_080304.pdf*

El-Sayed, A., Roca, V. & Mathy, L. (2003). A survey of proposals for an alternative group communication service, *IEEE Network* **17**(1): 46–51.

Finlayson, R. (2003). The UDP multicast tunneling protocol. IETF Draft <draft-finlayson-umtp-09.txt>.
URL: *http://tools.ietf.org/id/draft-finlayson-umtp-09.txt*

Francis, P. (2000). YOID: extending the internet multicast architecture. Unpublished report for ACIRI, `http://www.aciri.org/yoid/docs/index.html`.

Galvez, P. (2006). From VRVS to EVO (Enabling Virtual Organizations), *TERENA Networking Conference 2006*, Catania, Italy.

Hladká, E., Holub, P. & Denemark, J. (2004). An active network architecture: Distributed computer or transport medium, *3rd International Conference on Networking (ICN'04)*, Gosier, Guadeloupe, pp. 338–343.

Hladká, E., Liška, M. & Rebok, T. (2005). Stereoscopic video over ip networks, *in* P. Dini & P. Lorenz (eds), *International Conference on Networking and Services 2005 (ICNS'05)*, IEEE, p. 6.

Hladká, E. & Salvet, Z. (2001a). An active network architecture: Distributed computer or transport medium, *in* P. Lorenz (ed.), *Networking – ICN 2001: First International Conference Colmar, France, July 9-13, 2001, Proceedings, Part II*, Vol. 2094 of *Lecture Notes in Computer Science*, Springer-Verlag, Heidelberg, pp. 612–619.

Hladká, E. & Salvet, Z. (2001b). An active network architecture: Distributed computer or transport medium, *in* P. Lorenz (ed.), *Networking – ICN 2001: First International Conference Colmar, France, July 9-13, 2001, Proceedings, Part II*, Vol. 2094 of *Lecture Notes in Computer Science*, Springer-Verlag, Heidelberg, pp. 612–619.

Holer, V. (2003). *Slučování videostreamů (Videostream Merging)*, Bachelor thesis, Faculty of Informatics, Masaryk University Brno, Brno, Czech Republic.

Holub, P. (2005). *Network and Grid Support for Mulitmedia Distribution and Processing*, PhD thesis, Faculty of Informatics, Masaryk University Brno, Czech Republic.

Holub, P. & Hladká, E. (2006). Distributed active element for high-performance data distribution, *NPC 2006: Network and Parallel Computing*, Tokio, Japan, pp. 27–36.

Holub, P., Hladká, E. & Matyska, L. (2005). Scalability and robustness of virtual multicast for synchronous multimedia distribution, *Networking - ICN 2005: 4th International Conference on Networking, Reunion Island, France, April 17-21, 2005, Proceedings, Part II*, Vol. 3421/2005 of *Lecture Notes in Computer Science*, Springer-Verlag Heidelberg, La Réunion, France, pp. 876–883.

Holub, P., Hladká, E., Procházka, M. & Liška, M. (2007). Secure and pervasive collaborative platform for medical applications, *Studies in Health Technology and Informatics* **126**: 229–238. IOS Press, Amsterdam, The Netherlands.

Holub, P., Matyska, L., Liška, M., Hejtmánek, L., Denemark, J., Rebok, T., Hutanu, A., Paruchuri, R., Radil, J. & Hladká, E. (2006). High-definition multimedia for multiparty low-latency interactive communication, *Future Generation Computer Systems* **22**(8): 856–861.

Jannotti, J., Gifford, D. K., Johnson, K. L., Kaashoek, M. F. & O'Toole, J. J. W. (2000). Overcast: reliable multicasting with on overlay network, *OSDI'00: Proceedings of the 4th conference on Symposium on Operating System Design & Implementation*, USENIX Association, Berkeley, CA, USA, pp. 197–212.

Jo, J., Hong, W., Lee, S., Kim, D., Kim, J. & Byeon, O. (2006). Interactive 3D HD video transport for e-science collaboration over UCLP-enabled GLORIAD lightpath, *Future Generation Computer Systems* **22**(8): 884–891.

Li, Z. & Shin, Y. (2002). Survey of overlay multicast technology. Unpublished report for Networks Lab, UC Davis, `http://networks.cs.ucdavis.edu/~lizhi/papers/ECS289.pdf`.

Liebeherr, J. & Nahas, M. (2001). Application-layer multicast with delaunay triangulations, *Global Telecommunications Conference, 2001. GLOBECOM '01. IEEE*, pp. 1651–1655 vol.3.

Liška, M. & Denemark, J. (2006). Real-time transcoding and scalable data distribution for video streaming, *Technical Report 30/2006*, CESNET, Praha, Czech Republic.
URL: *http://www.cesnet.cz/doc/techzpravy/2006/rum-streaming/*

Liška, M. & Holub, P. (2009). Couniverse: Framework for building self-organizing collaborative environments using extreme-bandwidth media applications, *Euro-Par 2008 Workshops - Parallel Processing. Las Palmas de Gran Canaria, Spain*, Vol. 5415 of *Lecture Notes in Computer Science*, Springer-Verlag, Berlin/Heidelberg, pp. 339–351.

Mathy, L., Canonico, R. & Hutchison, D. (2001). An overlay tree building control protocol, *Lecture Notes in Computer Science* **2233**: 76–87.

Mathy, L., Canonico, R., Simpson, S. & Hutchison, D. (2002). Scalable adaptive hierarchical clustering, *NETWORKING '02: Proceedings of the Second International IFIP-TC6 Networking Conference on Networking Technologies, Services, and Protocols; Performance of Computer and Communication Networks; and Mobile and Wireless Communications*, Springer-Verlag, London, UK, pp. 1172–1177.

Matyska, L., Hladká, E. & Holub, P. (2007). Collaborative framework for distributed distance learning, *Proceedings of the 8th International Conference on Information Technology Based Higher Education and Training (ITHET '07)*, Kumamoto, Japan.

Matyska, L., Holub, P. & Hladká, E. (2007). Collaborative framework for distributed distance learning, *Proceedings of the 4th High-End Visualization Workshop*, Obergurgl, Tyrol, Austria, pp. 40–45.

McCanne, S. & Jacobson, V. (1995). vic: A flexible framework for packet video, *Proceedings of ACM Multimedia'95*, San Francisco, CA, USA, pp. 511–512.

Parnes, P., Synnes, K. & Schefstrom, D. (1998). Lightweight application level multicast tunneling using mtunnel, *Computer Communication* **21**(15): 1295–1301.

Procházka, M., Holub, P. & Hladká, E. (2005). Active element network with P2P control plane, *IWSOS 2006: 1st International Workshop on Self-Organizing Systems*, Vol. 4124/2006 of *Lecture Notes in Computer Science*, Springer-Verlag Heidelberg, University of Passau, Germany, p. 257.

Salvet, Z. (2001). Enhanced UDP packet reflector for unfriendly environments, *Technical Report 16/2001*, CESNET, Praha, Czech Republic.
URL: *http://www.cesnet.cz/doc/techzpravy/2001/16/*

Schulzrinne, H., Rao, A. & Lanphier, R. (1998). Real Time Streaming Protocol (RTSP). RFC 2326.

Shimizu, T., Shirai, D., Takahashi, H., Murooka, T., Obana, K., Tonomura, Y., Inoue, T., Yamaguchi, T., Fujii, T., Ohta, N., Ono, S., Aoyama, T., Herr, L., van Osdol, N., Wang, X., Brown, M. D., DeFanti, T. A., Feld, R., Balser, J., Morris, S., Henthorn, T., Dawe, G., Otto, P. & Smarr, L. (2006). International real-time streaming of 4K digital cinema, *Future Generation Computer Systems* **22**(8): 929–939.

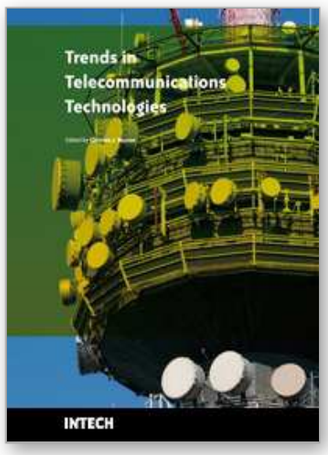
Thies, W., Karczmarek, M. & Amarasinghe, S. (2002). StreamIt: A language for streaming applications, *Proceedings of the 11th International Conference on Compiler Construction*, Vol. 2304/2002 of *Lecture Notes in Computer Science*, Springer-Verlag Heidelberg, Grenoble, France, pp. 179–196.

Tran, D. A., Hua, K. A. & Do, T. T. (2003). ZIGZAG: An efficient peer-to-peer scheme for media streaming., *INFOCOM*.

Zhang, B., Jamin, S. & Zhang, L. (2002). Host multicast: A framework for delivering multicast to end users., *IEEE INFOCOM '02*, New York, NY.

Zhuang, S. Q., Zhao, B. Y., Joseph, A. D., Katz, R. H. & Kubiatowicz, J. D. (2001). Bayeux: an architecture for scalable and fault-tolerant wide-area data dissemination, *NOSSDAV '01: Proceedings of the 11th international workshop on Network and operating systems support for digital audio and video*, ACM Press, New York, NY, USA, pp. 11–20.

IntechOpen



Trends in Telecommunications Technologies

Edited by Christos J Bouras

ISBN 978-953-307-072-8
Hard cover, 768 pages
Publisher InTech
Published online 01, March, 2010
Published in print edition March, 2010

The main focus of the book is the advances in telecommunications modeling, policy, and technology. In particular, several chapters of the book deal with low-level network layers and present issues in optical communication technology and optical networks, including the deployment of optical hardware devices and the design of optical network architecture. Wireless networking is also covered, with a focus on WiFi and WiMAX technologies. The book also contains chapters that deal with transport issues, and namely protocols and policies for efficient and guaranteed transmission characteristics while transferring demanding data applications such as video. Finally, the book includes chapters that focus on the delivery of applications through common telecommunication channels such as the earth atmosphere. This book is useful for researchers working in the telecommunications field, in order to read a compact gathering of some of the latest efforts in related areas. It is also useful for educators that wish to get an up-to-date glimpse of telecommunications research and present it in an easily understandable and concise way. It is finally suitable for the engineers and other interested people that would benefit from an overview of ideas, experiments, algorithms and techniques that are presented throughout the book.

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen