Active Element Network with P2P Control Plane

Michal Procházka[1,3], Petr Holub[1,3], Eva Hladká[2,3]

[1]Institute of Computer Science and [2]Faculty of Informatics,
Masaryk University, Botanická 68a, 602 00 Brno, Czech Republic

[3]CESNET z. s. p. o., Zikova 4, 160 00 Praha 6, Czech Republic

e-mail: {michalp,hopet}@ics.muni.cz,eva@fi.muni.cz

Abstract

Multi-point data distribution for synchronous multimedia communication poses interesting problem for networking environment usually implemented by either native or virtual multicast. In this paper, we describe and evaluate a scalable network of Active Elements (AE) that implements user-empowered virtual-multicast overlay network for synchronous data distribution and processing in the network. The AE network is based on strict separation of control plane and data plane. The control plane is organized as peer-to-peer network in order to achieve robustness and user-empowered approach while sacrificing efficiency to some extent. The data plane which handles the actual data distribution is optimized for efficiency and allows pluggable implementation of different distribution models. We present a prototype implementation with control plane based on JXTA peer-to-peer substrate and evaluate its behavior, robustness, and efficiency. We also describe some pilot applications the AEs network can be used with.

1 Introduction

Virtual collaborative environments as well as general multi-point synchronous multimedia distribution require some kind of multi-point distribution service. This can be implemented either as a native network service (e.g. IPv4 or IPv6 multicast), or it needs to be implemented on application level. In our previous work, we have shown both advantages and drawbacks of implementing this service on application level based on UDP packet reflector [1]. This approach has been later generalized into Active Element (AE), which supports building AE networks for more scalable and robust data distribution [2]. The AE is based on strict separation of control plane used for network setup and maintenance and data distribution plane used for the actual data distribution to clients. The AEs have shown to be useful for a wide scale of applications, ranging from low-bandwidth streams for mobile devices to high-definition uncompressed video streams with multi-gigabit bandwidth [3]. In this paper, we focus on implementing a peer-to-peer (P2P) organized control plane based on JXTA P2P framework [4] and evaluating its behavior, performance, and robustness.

Related work There is significant effort put into building P2P networks and frameworks. Most of them are however used primarily for asynchronous applications like file sharing. A synchronous data distribution P2P network similar to our AEs is EVO [5], announced about a year after our AE approach was published. This project tries to augment VRVS videoconferencing service [6] based on reflectors deployed in centralistic way. EVO adds intelligent overlay network on the top of the existing network of reflectors. This overlay network comprises Panda agents, which operate reflectors, and Koala end-user Java based application. Compared to our system, it is provided as closed source software bundle. Another synchronous multimedia distribution application based on P2P approach is Skype, but this is closed-source and little is known about its actual internals, as the authors have made it very obfuscated as shown in [7].

The rest of the paper is organized as follows: Section 2 briefs architecture of AE with separated data plane and control plane and discusses interactions within an AE network. Section 3 gives description of the actual implementation called jSon and results of the prototype evaluation are discussed in Section 3.2. Section 4 proposes steps for future work and work summary and concluding remarks are given in Section 5.

2 Network of Active Elements

The Active Element (AE) [1] is a user-empowered UDP packet reflector, which replicates all incoming traffic to all registered clients. The term user-empowered means that the whole environment can be set up and managed by a user and there is no need for additional network services nor special support by system/network administrators. Because all data goes through the AE, the AE can process it in various ways. Thus the AE can ensure multipoint data distribution and on-demand processing for user-empowered collaborative environments. The serious problem of the single AE is limited scalability and missing robustness and therefore AE network has been proposed. The first solution were AEs interconnected by predefined static data distribution models [8]. But such a system lacks load-balancing, dynamic recovery capability, and it requires a manual setup.

Therefore the model has been extended into a self-organizing AE network [2] that features higher level of robustness and scalability for the collaborative environment support. The self-organized AE network assumes strict separation of control plane and data plane. The control plane is used for building control channels between the AEs and also for distributing information on available collaborative groups, content, capabilities of each individual AE, and also properties of the underlying network. The control plane is supposed to be implemented as a peer-to-peer overlay network in order to achieve maximum robustness while sticking to the user-empowered paradigm. The data plane is used for the actual synchronous distribution of multimedia content distributed by the AE network. It allows for pluggable implementation of various data distribution models, which can be deployed based on their suitability to specific environment, where the AE network is run.

The level of robustness and scalability of the AE network depends on both the control plane and the data plane. For the robustness, the control plane is however more important, as the data distribution model can be recomputed or even changed to different model, provided that control plane is able to keep communication channels between the AEs established. On the other hand, scalability of the AE network depends on the data distribution model, as the amount of communication through the control plane is much lower compared to the data plane.

Data Distribution Plane Data distribution models specify data distribution among AEs and clients in the specific collaborative session. They need to support fast rebuilding in case of failure, based on information provided by the control plane. Also they need to adapt to largely varying requirements of client applications – some may be critical in terms of latency, others in terms of bandwidth utilized. Several basic distribution models have been described and analyzed in terms of scalability and robustness in [2], including simple 2D full mesh model, 3D layered mesh model with and without intermediate AEs, and (multiple redundant) spanning trees.

Peer-to-Peer Control Plane The control plane for the AE network has been proposed to be of P2P architecture because of the following three reasons: P2P networks (1) provide auto-discovery mechanisms so that new network elements can join the network without complicated pre-configuration by the user, (2) feature very good degree of robustness under changing network conditions and even in hostile networking environments including firewalls and NATs, (3) retain user-empowered approach entirely, so that no administrative permissions are needed for their setup and operation. Because the P2P network route messages inside the P2P overlay network, the message does not need to follow the shortest path in the network – but this is not significant problem for the control messages, unless the latency increase compared to the underlying network is exceptionally large.

P2P networks can be divided into the several categories, such as pure and hybrid P2P architectures. We have chosen hybrid P2P topology with loosely-consistent distributed hash table (DHT) search mechanism because it known to have balanced levels of scalability and robustness. It also allows searching for known content (hash search) as well as for unknown content (wildcard search). Hybrid type of P2P needs fewer management messages to get working compared to pure P2P and simple DHT-based models.

Communication in the AE network There are two types of entities in the AE networks: AEs and clients. The AEs manage the whole AE network and distribution schemes, while clients may only join the network, look for content, create/destroy conferences, join/leave the conferences, and send/receive data. For robustness reasons, each AE and also each client may have multiple (redundant) connections established with other AEs.

Clients are able to create conferences by making a request to the AE network and any suitable AE handles the request (anycast approach). This AE creates the group, which represents the conference, and invites the client to it. Then the communication tunnel may be established for audio, video, or other type of media among the AE and the client. In terms of the distribution scheme, it invites other AEs into this newly created group. The distribution schemes are computed and applicable only within this group.

The conference creating request, which user sends to the AE, contains information about conference itself such as name, description, start of conference, end of conference, and access rules. The conferences are persistent for the time given in the conference initiation request. This means that a conference exists even if there are no clients connected to it; this feature can be used, e.g. when conference participants need to store some data persistently for the whole duration of the conference regardless of whether clients are connected or not.

When the conference is created, anybody can search for that conference and if the access rules allow it, he/she can join the conference. The client and the AE periodically exchange information about AEs participating in the current videoconference. AE also periodically monitors the distance between his neighbors based on round-trip time (RTT). It can be also extended to support other measures like available bandwidth, jitter, and other parameters important for synchronous multimedia distribution and processing. This behavior is important for the data distribution plane to be optimized, e.g. to compute minimum spanning tree, the control plane provides the graph comprising AEs as nodes and network lines as RTT-weighted vertices.

When some failure occurs in the network or some AE fails while the conference is running, the control plane of the AE network recognizes it and initiates reorganization of the actual distribution scheme in a time-frame of seconds.

3 jSon Prototype

The prototype implementation of the AE-based with JXTA-based control plane has been named jSon by us. It is a stand-alone modular application, which is split into the following two parts: the first part provides functionality of the AE and the second part implements the client side. Example of an AE network using jSon is shown in the Figure 1.

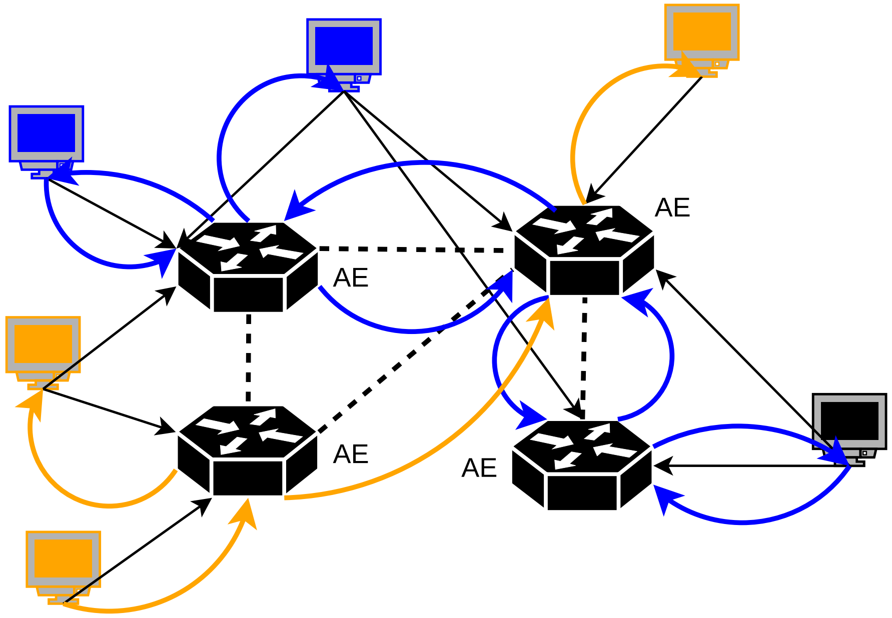

Figure 1: Example of a jSon network. Communication between clients and AEs is shown in the black lines (⟶), while inter-AE communication is shown in dotted black lines (····). Colored lines (→, →) represent data flows in two collaborative sessions.

3.1 Prototype Implementation

For implementing jSon we chose JXTA P2P framework [4]. The JXTA is a set of protocols which can create a virtual overlay network on top of the existing network infrastructure. The JXTA virtual network allows peers to exchange messages with other peers independently of their network location and trickiness of the network environment. Messages are transparently routed, potentially traversing firewalls and/or NATs, possibly using different transport/transfer protocols (TCP/IP, HTTP, etc.) to reach the receiving peers even if these are on non-IP networks.

jSon is implemented on top of a JXTA implementation in the C language (JXTA-C), because the whole AE framework has been implemented in the C language previously for portability and efficiency reasons. We had to do some changes to the JXTA framework to better fit our needs. We had to fix numerous bugs[1], added functionality like is getting bootstrap peers from the web servers, improved logging facility to provide information needed for evaluating robustness and scalability, and also added support for JxtaNetMap protocol. We also changed built-in parameters of JXTA to react faster on failures occurring in the AE network as discussed in last part of performance evaluation below.

jSon is a modular application consisting of the following modules:

- *ID generation* – Each entity in JXTA network has its own identity called ID. This module can override JXTA default behavior of ID generation.

- *Network Information Service* – Information about available content and capabilities of the AE is managed by this module and published into the AE network. This module gathers information like machine hostname and underlying network address, operating system, capabilities of AE, and many others. It also maintains information on available content – currently running collaborative sessions together with available formats of media/data streams, all maintained in hierarchical structure described in [2].

- *Messenger center* – This module takes care of messages handling and consists of two parts – one for sending messages and the other for receiving messages. The messages may be sent to all the peers within the initial group, any subgroup, or to particular peer or subset of peers. The messages have their own special format with each message having its own name, priority and parameters. Priority determines the order of the processing of messages received at the same time on the receiving side and it determines the type of the message, too. If the priority is less than one thousand, the message is control message; if the message priority is greater then one thousand, it is

[1] JXTA implementation in C seems to be less developed and much less maintained compared to the reference implementation in Java.

an informational message. On a reception event, the message is parsed by this module and data is sent to the appropriate function for further processing.

- *Monitor center* – This module is used for gathering information necessary for JXTA-NetMap [9] application that is a network visualization tool for JXTA networks. We have special requirements on the monitoring and therefore we have introduced small changes to the JXTANetMap protocol. Thus, it is now possible to display various state information about the AE such as connected clients, uptime of the AE, available bandwidth for the AE data plane, etc.
- *Group management* – manages peer groups. The groups are being founded dynamically on demand, for example "the group of AEs that supports some transcoding". Each peer group is described by a structure which contains all the necessary information to identify the group, to get the name and ID, to get the description of the group, and to get the list of peers in this group.
- *Discovery* – This module handles all discovery service messages. It can react on each incoming message. When a peer is searching for some unknown information in the network (e.g. a wildcard search), it sends a discovery message. This message is propagated within peer group and every peer can react on this message using Discovery module. This module also handles peer discovery in the network, e.g. when the jSon-based AE joins in the existing AE network. Group management module utilizes Discovery module for peer list updating.
- *Diagnostic* – is only used for debugging and testing purposes. It contains a set of testing functions for message sending, polling information about peer status, about connected peers, and about actual state of peer view.

3.2 Prototype Evaluation

In order to evaluate prototype behavior of the P2P control plane in real conditions, we made a series of tests limited to the jSon and the JXTA network. We do not evaluate scalability and robustness of the data distribution, as this has already been studied previously in [2].

Our testbed infrastructure consisted of eight computers with single Pentium 4 processor and 512 MB memory. All the computers were connected to a local 100 Mbps flat switched network.

Scalability The first set of tests was focused on P2P control plane scalability. We have evaluated the influence of growing number of the AEs in the network on the total number of the management messages in the AE network and on the number of the management messages sent and received by an individual AE as shown in the Figure 2. We have started with a network of two AEs and after each approximately 20 minutes, another AE joined this network (marked with red dashed line). As we can see from the graph, the steady-state number of the messages in the whole network is growing nearly linearly with respect to the number of AEs. The peeks in the graph represent information exchange which JXTA is doing in order to reorganize itself. Also when examining the number of messages sent and received by an individual AE during the same test, it turns out that it remains almost constant (Figure 3). This means that the growing number of AEs in the network does not have any significant influence on the load on the other AEs and therefore the scalability of the AE network is indeed limited by the data distribution model and amount of data sent in the network.

To assess long-term stability of the jSon control messaging, we also made long term test where eight AEs were connected to the AE network for forty hours. During the test, AEs were left without any intervention. The plot in Figure 4 shows number of messages exchanged among AEs in ten minutes interval. We can see that after initial message exchange, the number of management messages is stabilized.

Robustness As discussed in the Section 2, a good level of robustness of the AE network control plane is crucial requirement. It is critical for the data plane robustness since the data plane reorganization depends on the input information provided by the control plane. Because each AE periodically monitors the availability of its neighbors by the means of JXTA framework, it can detect the failure of neighbor AE or failure of link between them in maximum timeout which is specified in the configuration of each AE. That is why the robustness

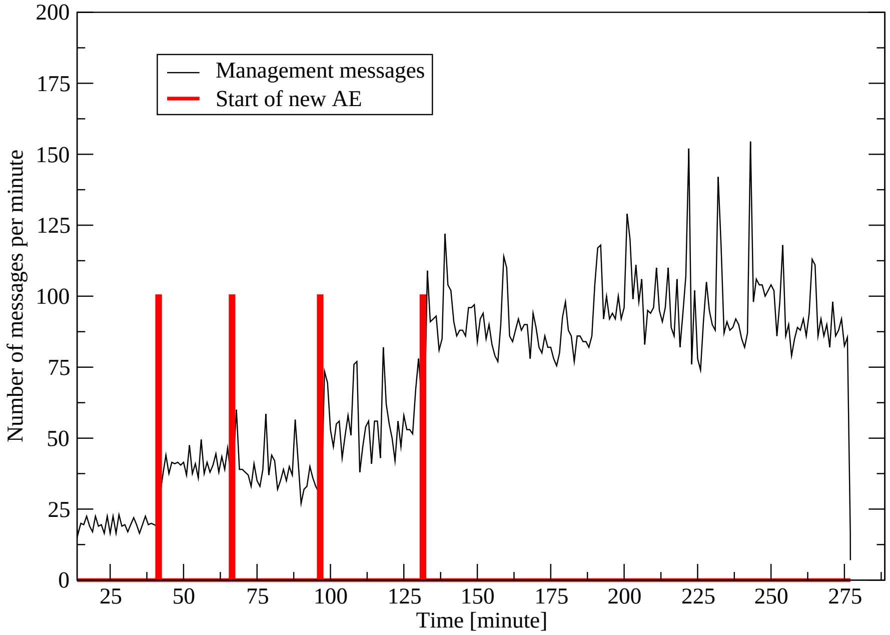

Figure 2: Influence of growing number of peers on a number of management messages

evaluation of jSon is actually an evaluation JXTA network.

The test consists of eight AEs that continuously exchange messages and these messages are gathered at each AE. We chose one AE which was connected (column *Start AE* in Tables 1 and 2) and disconnected (column *Stop AE*) from the AE network in defined times. Time in the table is related to the start of the test. The column *Start receiving msg* indicates the time when the connected AE was starting to receive messages. The column *Delay* indicates the time difference between the connection time and time when AE starts receiving messages.

We made two tests, the first with default setup of JXTA-C and second with tuned parametrization . As shown in Table 1, the default setup of JXTA-C results in a big delay. This is not suitable for robust synchronous collaborative environment, where reactions on failures are needed as quick as possible. Therefore we have introduced changes in parametrization of the JXTA framework: (a) we have increased the number of concurrently connected neighbors from 1 to 3, (b) decreased the interval between searching for new AEs, when the AE is not connected, from 2 seconds to half of a second, and (c) decreased the interval for which the peer is marked as unreachable, once its unreachability is detected, from 10 minutes to 5

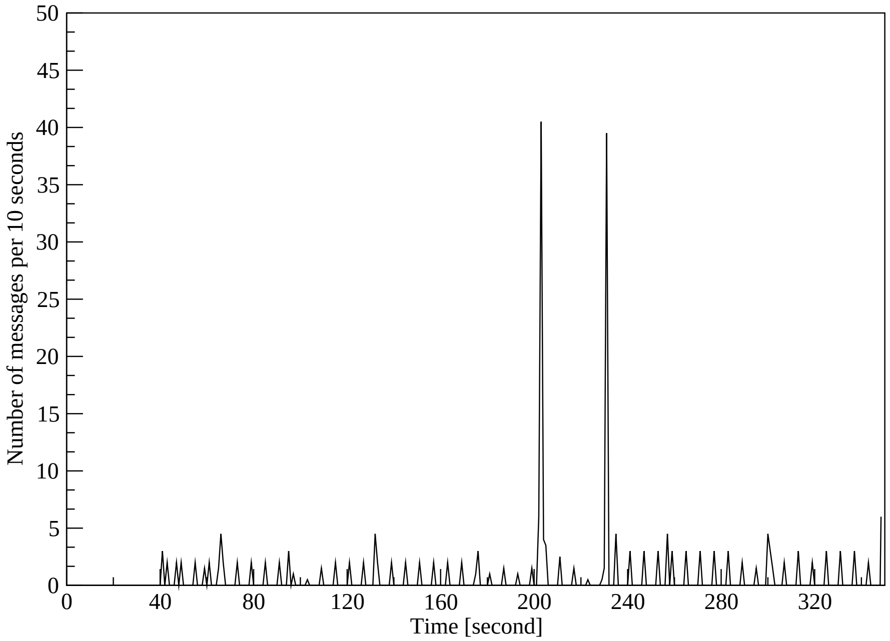

Figure 3: Influence of growing number of peers on individual AE

seconds. These changes have resulted in shortening the AE reconnection time into the range of seconds (generally under one second) as shown in Table 2).

Round	Stop AE [s]	Start AE [s]	Start receiving msg [s]	Delay [s]
1.		61	62	
2.	98	154	220	66
3.	267	273	337	64
4.	355			

Table 1: Default JXTA-C set up

Latency We have also evaluated behavior of the JXTA compared to the underlying network in terms of RTT using standard IPv4 ping and JXTA ping. Though latency is critical in the data plane and not in the control plane, it is a useful measure of "efficiency" of an overlay P2P network. As we can see in the Figure 5, there was an average increase about 100 μs compared to underlying network. This measurement has however very limited significance, as it was measured in the non-structured switched network, where communicating nodes did not have any reason to communicate indirectly as in networks with complex topologies.

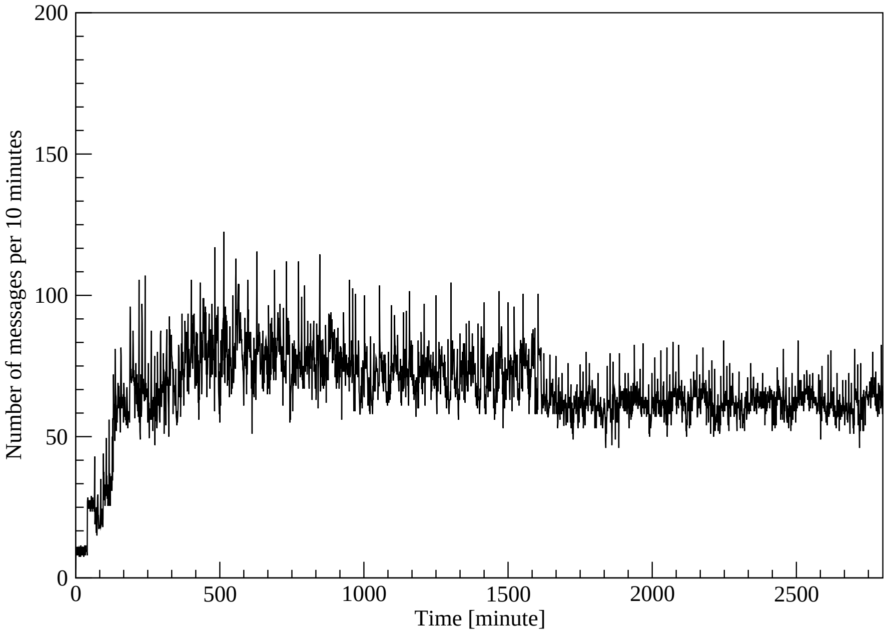

Figure 4: Long term measure of number of management messages

Round	Stop AE [s]	Start AE [s]	Start receiving msg [s]	Delay [s]
1.		48	49	
2.	83	121	122	1
3.	161	183	184	1
4.	215	240	241	1
5.	263	275	276	1
6.	300			

Table 2: Modified JXTA-C set up

3.3 Prototype Applications

The AEs networks are designed to be used with generally any multimedia application, that relies on RTP/UDP data transmission. This includes both videoconferencing tools like MBone Tools [10] (RAT for audio, VIC for video, WB/WBD for shared whiteboard) or OpenMash [11] and unidirectional "broadcasting" applications like VideoLAN Client [12].

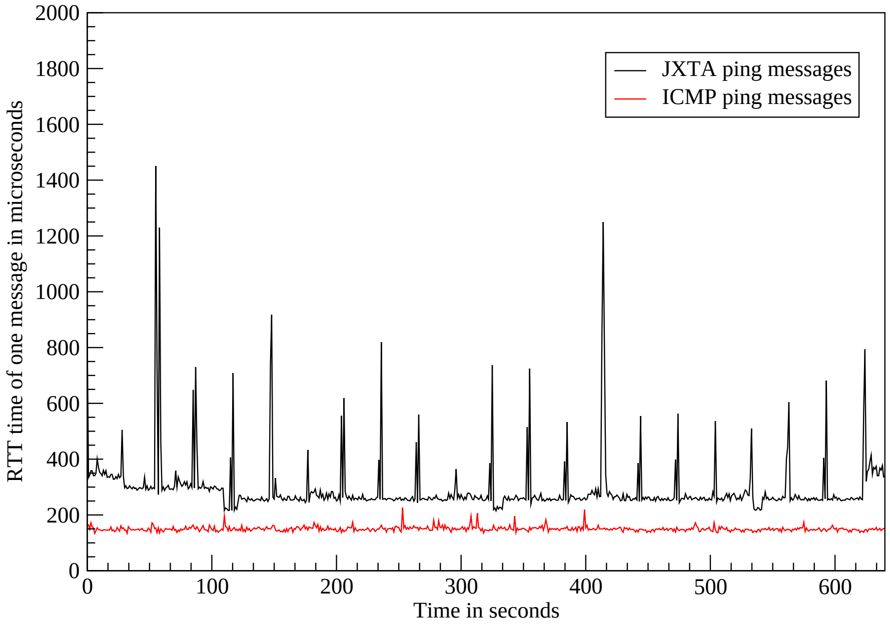

Figure 5: Difference between IPv4 ping and JXTA ping

4 Future work

Our main goal is to finish production implementation of jSon. We hope the JXTA-C will be also finished soon because we had to solve a number of problems with JXTA-C as mentioned above. We also considering reimplementing the control plane in Java and connect it to the AE framework using Java Native Interface (JNI) and Reflector Administration Protocol (RAP) [13].

In the future we would like to generate IDs on the basis of locality in the ID generation module. There are two solutions how to handle the locality problem. First of all we can follow pure DHT networks searching mechanism and generate IDs for closer entities (in the sense of physical distance) with similar prefix as long as possible. The second solution is to implement some naming service and adapt ID generating to this service.

Another area were we would like to be involved in is security, especially secure AE network join and secure message transport. We would also like to implement AAA (Authentication, Authorization, Auditing) into the jSon. Developing a suitable GUI (Graphic User Interface) application on the client side will hopefully make adoption of AE with jSon faster.

5 Conclusions

Active Elements have shown to be effective user-empowered solution for synchronous distribution of multimedia data in applications like videoconferencing and broadcasting. In this paper, we have described an implementation of P2P based control plane for Active Element. We presented the prototype implementation called jSon built on the top of the JXTA P2P framework and evaluated its performance in terms of scalability and robustness. Scalability evaluation confirms, that scalability is not limited by the control plane of the AE network. Robustness of the JXTA-based network is good, but to achieve recovery times suitable for synchronous communication, re-parametrization of the JXTA was necessary. On a flat switched network topology, latency increase for message passing in the P2P overlay network is not significant compared to the latency of the underlying network.

Acknowledgments

This work has been supported by a research intent "Optical Network of National Research and Its New Applications" (MŠM 6383917201) and "Parallel and Distributed Systems" (MŠM 0021622419).

References

[1] Eva Hladká, Petr Holub, and Jiří Denemark. An active network architecture: Distributed computer or transport medium. In *3rd International Conference on Networking (ICN'04)*, pages 338–343, Gosier, Guadeloupe, March 2004.

[2] Petr Holub, Eva Hladká, and Luděk Matyska. Scalability and robustness of virtual multicast for synchronous multimedia distribution. In *Networking - ICN 2005: 4th International Conference on Networking, Reunion Island, France, April 17-21, 2005, Proceedings, Part II*, volume 3421/2005 of *Lecture Notes in Computer Science*, pages 876–883, La Réunion, France, April 2005. Springer-Verlag Heidelberg.

[3] Petr Holub, Luděk Matyska, Miloš Liška, Lukáš Hejtmánek, Jiří Denemark, Andrei Hutanu, Ravi Paruchuri, Jan Radil, and Eva Hladká. High-definition multimedia for multiparty low-latency interactive communication. *Future Generation Computer Systems*, 2006. *Accepted.*

[4] Bernard Traversat, Ahkil Arora, Mohamed Abdelaziz, Mike Duigou, Carl Haywood, Jean-Christophe Hugly, Eric Pouyoul, and Bill Yeager. Project JXTA 2.0 super-peer virtual network. `http://www.jxta.org/project/www/docs/JXTA2.0protocols1.pdf`.

[5] Evo - end-to-end self managed rtc infrastructure, November 2005. `http://monalisa.cern.ch:8080/Upload/VRVS-EVO_ESnet05.ppt`.

[6] Vrvs - virtual room videoconferencing system, Decmber 2005. `http://www.vrvs.org/index.php`.

[7] Philippe Biondi and Fabrice Desclaux. Silver needle in the skype. In *BlackHat Europe*, March 2006. `http://www.secdev.org/conf/skype_BHEU06.handout.pdf`.

[8] Eva Hladká, Petr Holub, and Jiří Denemark. User empowered programmable network support for collaborative environment. In *ECUMN'04*, volume 3262/2004 of *Lecture Notes in Computer Science*, pages 367 – 376. Springer-Verlag Heidelberg, 2004.

[9] Mohamed Abdelaziz, Jean-Christophe Hugly, Dave Bryson, and Mathieu Jan. Project JXTA NetMap, 2006. `http://jxtanetmap.jxta.org`.

[10] MBone tools. `http://www-mice.cs.ucl.ac.uk/multimedia/software/`.

[11] Open Mash project. `http://www.openmash.org/`.

[12] VideoLAN Client (VLC). `http://www.videolan.org/`.

[13] Jiří Denemark, Petr Holub, and Eva Hladká. RAP – reflector administration protocol. Technical Report 9/2003, CESNET, 2003.