Proceedings of the 8th IEEE Symposium on Computers and Communications - ISCC'2003, Antalya, Turkey, June 2003 June 2003, pp.1435-1440., IEEE Computer Society Press (unformatted version)

Symbiotic Streaming of Elastic Traffic on Interactive Transport

Javed I. Khan, Raid Zaghal, and Qiong Gu
Networking and Media Communications Research Laboratories
Department of Computer Science, Kent State University
233 MSB, Kent, OH 44242
javedlrzaghal@cs.kent.edu

Abstract

Interactivity in transport protocol can greatly benefit transport friendly applications. We envision if a transport mechanism, which is interactive and can provide event notification about network state to the subscriber of its communication service, than a wide range of solution to many of today's hard network problem can be instituted. Recently we have implemented this concept system as a new TCP kernel on FreeBSD called TCP Interactive and a novel symbiotic MPEG-2 full logic transcoder. In this paper we share the results of the TCP Interactive performance experiment and show potential dramatic improvement in time-bounded video delivery.

Index terms-- netcentric applications, TCP interactive, transcoding, temporal QoS.

1. Introduction

Recently, TCP-friendliness has been suggested applications for congestion control [12]. However, a major problem in attaining any such friendliness is that the network itself extends very little help to the applications to encourage friendliness. A particular difficult problem is the congestion control for time-sensitive multimedia traffic. Most of the mechanisms for congestion control those have been proposed to date are based on delaying traffic at various network points. The more classical schemes depend on numerous variants of packet dropping in network, prioritization (graceful delay in router buffer) admission control (delaying at network egress points), etc. However, a key aspect of a vast majority of these schemes is that they introduce *time distortion* in the transport pathway. Though time distortion does no harm to time insensitive traffic, but they work completely against the application if the traffic is time sensitive such as multimedia streaming or control data. In this context, this paper presents a new approach. The suggestion does not require any new protocol per se but suggests adding interactivity with subscriber layer within current transport protocols. We then demonstrate effective solution to the hard problem of congestion control for time sensitive traffic based on this principle of protocol interactivity. The general principle we follow is simple and intuitive. It seems an effective delay conformant solution for time sensitive traffic may be built if the original data volume can be reduced by its originator-- the application. However, a key element in any such scheme is that the application must be notified. Unfortunately, today's transport protocols do not support any interactivity with applications. It seems such non-interactivity has been inherited from the early days of networking interface research, when the applications were simple[1]. In this paper we will show that transport interactivity can bring major benefit to high performance and demanding applications. The particular scheme we propose suggest a new paradigm of TCP friendliness based on enhanced network and its subscriber layer interactivity and has the following novel aspects:

- First, we suggest an active and direct notification mechanism by the underlying transport protocol, rather than using indirect end-to-end feedback tools.
- To demonstrate the efficacy of the principle, we have designed a corresponding video rate transcoder system that works in symbiosis with the network. This transcoder actively participates in a custom symbiotic exponential-back-off and additive-increase like scheme in application layer with deep application level knowledge. (This is also one of the first to our knowledge) resulting in much more effective joint quality/delay sensitive communication.
- The resulting scheme is similar in spirit to the TCP friendly approaches. However, there is a fundamental difference in how it is done. We expect network (or system) layers to remain as simple as possible. The means and techniques for rate reduction remain with the producer application. The responsibility of the network

[1] Interestingly the original TCP proposal (RFC007, RFC793) did call for asynchronous interlayer interactivity but it was never implemented. Need for some forms of callback facility have been marginally cited in recent literature, unfortunately without any technical depth. A survey is available at [13].

layer is simply to pass on only selected end-point events to the applications.

The adaptation is applicable for traffic where it is possible to dynamically adjust the data generation rate. We call it ***elastic traffic***. Most perceptual data, such as audio, video streams generally belongs to this traffic class. As, we will show the scheme is not only intuitive and simple, but also surprisingly effective compared to many other recently proposed schemes, which involve much more complex system/network layer reorganization.

The result presented in this paper is not simulation; rather report from a real implementation of the concept system that we have completed very recently. The implementation has two components-- an interactive transport over FreeBSD and, a novel symbiotic MPEG-2 full logic transcoder [6], which is capable of working in tandem with the interactive transport.

In the next section, we provide the symbiosis system overview. In section 3 we share the experiment results and the performance of the scheme. We have experimented the system in a controlled network with various levels of injected congestion. More background information and related work can be found in [7].

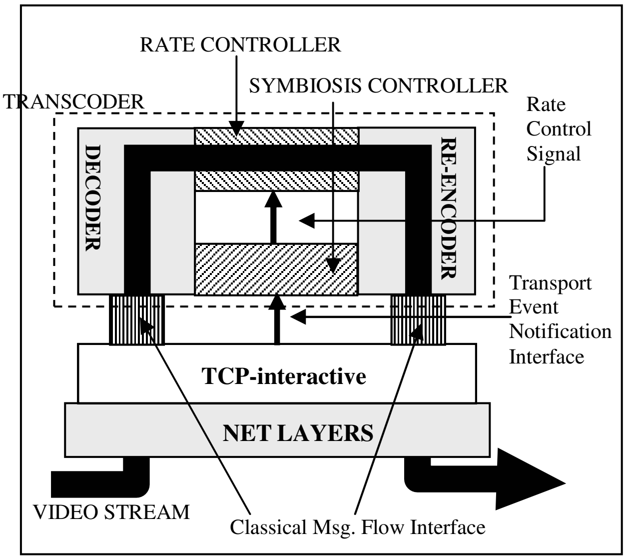

Fig-1. Interactive transport and symbiotic transcoder.

2. Rate Adaptive Transcoder

2.1. System Configuration

We have developed a three-part system model-- *server, transcoder* and *the client*. The middle component *transcoder* [8, 5] can be placed in a suitable network junction point, which intercepts the stream. This is slightly different from encoder-decoder (*server-client)*

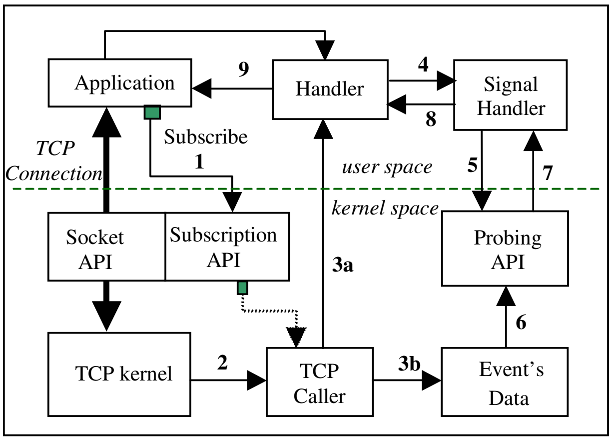

Fig-2. The TCP-interactive extension. The added registration API allows demanding applications to subscribe to events and probe additional event data.

system model. The system has been designed to sit either at the transport entry-point and perform conventional end-to-end paradigm based video transport like a conventional encoder. Or, it can also sit inside a network using technology such as a proxy for targeted and localized congestion management. This approach has several advantages. It subsumes *(i)* the functionalities of server-client model. In addition, *(ii)* it allows rate adaptation on video stream that is already encoded and thus enables serving stored video at a dynamically selected rate. *(iii)* The decoupling also has the benefit that the transcoder can be made to auto sense local asymmetry in link capacities and can be dynamically deployed inside network for streaming. For example it can sit at a node splicing a fiber and a wireless network, and thus can downscale an incoming high-bandwidth video multicast stream for an outgoing low-capacity wireless links using techniques such as [10, 1].

2.2. Transport Control

The transcoder sits on top of the interactive transport control layer-- TCP Interactive. Fig-1 explains the system arrangement. Unlike conventional TCP, this interactive transport layer, when there is an internal timer-out event, passes on the current window resize event to the subscriber layer. The interface is almost identical to the TCP classic. There is no difference for classical applications. However, upon opening the socket, an adaptive application may bind an interrupt handler routine to the designated socket event. When, the event occurs the TCP triggers the handler. The binding is optional. If application chooses not to bind any handler the system defaults to the silent mode identical to TCP classic.

The internal architecture of the TCP-interactive is shown in Fig-2. The added *subscription API* helps applications to

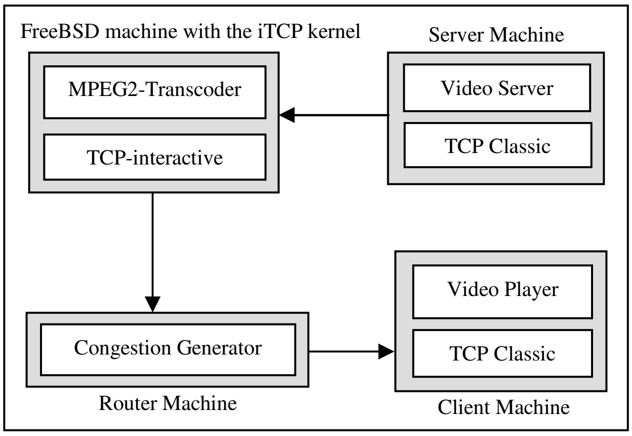

Fig-3. Experiment Setup.

subscribe to TCP events, in this case the timer out. We have added a simple extension to the TCP kernel. The main unit is called *TCP Caller* unit. It is activated if the application subscribes. It keeps track of the TCP timeout events. More inquisitive applications can also probe into selected TCP states. When a timeout event is detected the Kernel initiates 3a via TCP caller then invokes the handler. Optionally, applications can probe additional event data via signal handler/ and additional API (4,5,6,7,8,9). If event data is subscribed then 3b occurs concurrently with 3a. In the design of the TCP-interactive protocol we have retained three important protocol engineering principles namely *(i) network functional compliancy* (ii) *state-transition* compliancy, and (iii) *default-to-classical extension* interface model. These principles provide important advantages to the scheme. Its interface mechanism with IP layer remains functionally identical with TCP classic. Thus, it remains fully operational with all other currently deployed TCP remote-hosts. Secondly, its *state-transition* behavior also remains identical to that of TCP classic. Thus, all other network embedded transparent elements, which rely on certain assumptions about TCP behavior (such as congestion control schemes), will also not be affected. Thirdly, the '*default-to-classical'* extension of application programming interface enables the TCP-interactive host to concurrently run all other existing applications without any code modification. Thus, it keeps all legacy applications 100% compatible even on the updated host. Notably, many other suggested congestion management potentially violate these critical protocol extension principles.

3. Experiment Results

3.1. Setup

We have implemented an MPEG-2 DTV symbiotic video rate transcoder that uses the above model. We have also implemented a TCP-interactive transport on a FreeBSD system. Fig-3 shows the experiment setup. In this scheme a transcoder has been placed on an intermediate node between the server and the client. Between the transcoder and the client we have also placed a *Congestion Injector* to create variety of congestion situations for the video stream.

3.1.1. Congestion Injector. The Congestion Injector alters the routing table periodically. The congestion has been modeled as a sequence of bursts. The injector allowed the time between each consecutive burst and its duration to be individually specified. Two vectors *InterBurstTime*[], and *BurstDuration*[] are used for that purpose.

In our experiment, we used bursts with 3 seconds duration and random inter-burst time. To simulate various intensity of congestion we allowed the congestion to be increasingly more frequent within the transmission time of the test video. We let the congestion happen 3, 6 and 9 times within the transmission interval. However, the *InterBurstTime*[i] was generated randomly to avoid any effect of periodicity.

3.1.2. Traffic Characteristics. This experiment describes the performance for the case of a MPEG-2 ISO/IEC 13818-2 broadcast DTV (704x480) resolution video encoded with base frame rate of 2 Mbps at main level profile on this symbiotic transcoder.

For these set of experiments we run the transport subsystem to operate both in the classical transport mode (labeled as TCP) and in interactive transport mode (labeled as iTCP). We let the video generator (transcoder) feed into the video stream. In the classical mode, we switched off the improvements and let the transcoder operate in conventional error unaware mode. The transcoder generated the video using the conventional TM-5 [9] rate control at the target rate of 2 Mbps. Transport protocol buffered the generated data while the transport layer exercised binary back-off and additive recovery at time-out events. In the interactive mode, we switched on the interactive mechanism in the transport layer and the symbiosis mechanism of the transcoder. The transcoder according to the symbiosis controller varied the video rate for interactive TCP.

3.2. Symbiotic Rate Control

Fig-4 shows the symbiotic frame rate transcoding that occurred due to the joint rate specification at the rate control logic at the symbiosis unit in the transcoder. It plots the incoming video frame sizes, the target *rate retraction ratio* specified by the symbiosis controller, and the resulting outgoing frame rate generated by the transcoder. The timer out events (in this case there are 12 events generated in six bursts) at the TCP resulted in the symbiosis unit to modify the rate according to the lazy-

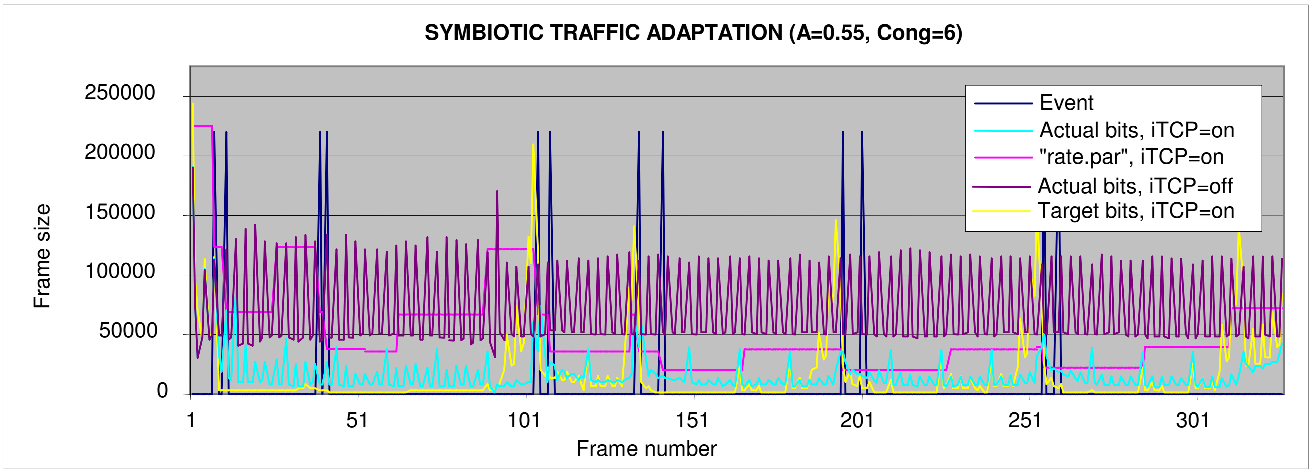

Fig-4. The binary-back off rate reduction in transcoder. There are six congestion bursts. The figure plots the incoming frame sizes, the event driven target rate (retraction ratio) specified by the symbiosis unit, and the resulting output rate from the transcoder.

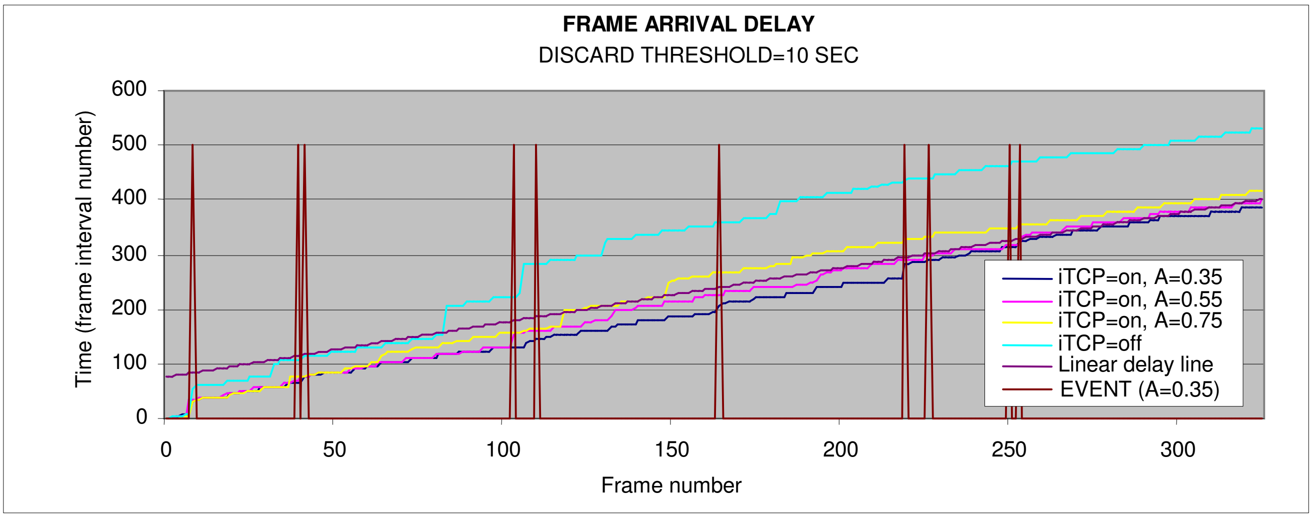

Fig-5. The figure plots the arrival time of the frames. For ideal case it should be linear. However, with each timeout event the backlog increased (observed as the step jumps in delay). The iTCP helps in eliminating these step increases in delays. The events correspond to A=0.35 experiment.

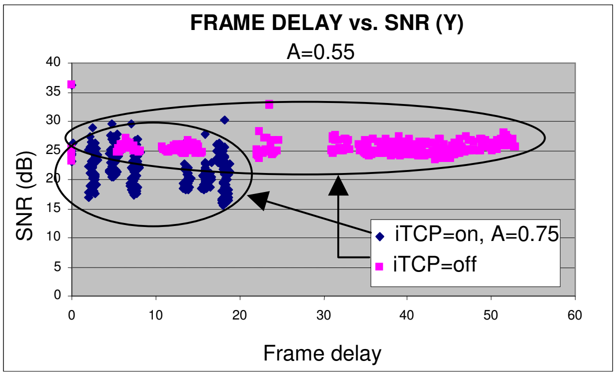

Fig-6. The two clusters show the quality, delay tradeoff offered by the iTCP. The iTCP dramatically reduced frame delivery delay by controlled trade-off of the SNR quality.

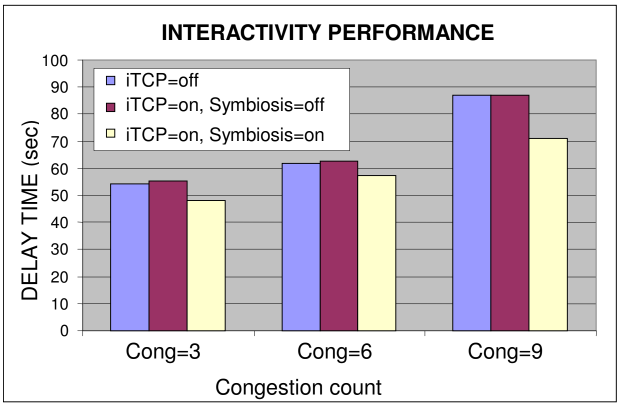

Fig-7. Compared to the overall transport time without interactivity (iTCP=off), there is a small overhead associated with iTCP event trapping mechanism (iTCP=on, Symbiosis=off). However, this small overhead is vastly outweighed by the symbiosis (iTCP=on, Symbiosis=on).

	Average Delay	SNR(Y)	SNR(U)	SNR(V)
iTCP=on, A=0.35	10.31	16.14	9.08	9.92
iTCP=on, A=0.55	12.81	15.61	8.67	9.46
iTCP=on, A=0.75	10.50	21.59	12.84	14.01
iTCP=off	32.27	25.31	15.90	17.21

Table-1 The average delay vs. picture quality

binary-back-off rule. A retraction ratio (A) of 0.55 was used. Though, the final generation rate varied widely from frame to frame due to their frame type, but the general trend followed the specified target.

3.3. MPEG-2 Frame Transport Efficiency

Now we show the impact of TCP interactivity. In the first experiment, we took frame wise detail event trace of what happens to the first 326 of the frames of this video at both sending and receiving ends. For a given discard threshold time in the receiving end we also traced which frame was successfully received or not at the receiving end of the MPEG-2 player. For comparison we traced both the transport unaware and transport aware cases. In Fig-5 we show the delay experienced by the video frames in terms of frame arrival interval. The figure plots the frame arrival time for classical TCP (iTCP=off), and three cases of interactive TCP (iTCP, A=0.35,0.55 and 0.75). For comparison in the graph we also show the ideal expected frame delivery time based on linear generation rate. As can be seen the iTCP outperformed the classical TCP. The figure also shows the events times (for the case of A=0.35). As it can be shown after each congestion burst, TCP continuously fell behind. The delay built up and hardly it could recover. This is shown by the step jumps in the delay line. The iTCP also suffered some step buildup, but it was much smaller and it could recover after few seconds.

3.4. Observation at Application Level

In the above two experiments we illustrated how the symbiosis mechanism worked from the video transport protocol (MPEG-2) and the network transport protocol (TCP) layers beneath it. In this plot we will illustrate how this mechanism appears from the very top-- in the application layer itself. An application receives and delivers uncompressed frames. The performance metric this end-system uses is the temporal and spatial quality difference between the transmitted and the reproduced uncompressed video frames at both ends. The underlying MPEG-2 protocol and the network layer TCP together provide the transport. The specific compression, windowing etc. and other detail mechanisms are external techniques to the end systems.

In Fig-6 each of the frames are plotted as a point in the video quality/frame delay plane. As can be seen from the region of the two QoS distributions, in classical TCP, although frames have been generated with SNR quality ranging between 22-29 dB, but many of these frames were lost in transport, and were never delivered. In contrast, the proposed TCP-interactive can deliver all the frames with 15-17 delay guaranteed at 15-26 dB quality[2]. This plots shows the Y block quality. Table-1 shows the average for all Y, U, and V blocks. Fundamentally, what TCP-interactive has offered is a qualitatively (as opposed to the quantitative improvements offered by any unaware solution) new empowering mechanism, where the catastrophic frame delay can be traded off for acceptable reduction in SNR quality.

3.5. Overhead of Event Service

The dramatic advantage in application level performance came at a cost since the event tracking mechanism added some overhead. We were also curious to find out the overhead of the event mechanism. To track the overhead, we recorded the total data transmission time under the following three conditions. First, we let the transcoder send the video over the classical TCP implementation of our BSD system. The left most bar of Fig-7 plots the transport time for the three congestion levels. To observe the overhead of the event service, in second run we used the iTCP implementation, however, we stopped the symbiotic reduction so the transport layer handled the same amount of data. As expected the overall transmission time increased in all three cases. However, in the third run we activated the event delivery and symbiosis. As can be seen, the slight increase in the event delivery overhead was vastly offset by the application level technique. The advantage the application gained from the event delivery was much bigger than the overhead.

4. Conclusions and Current Work

In this paper, we have presented a case of rate symbiosis mechanism in line with current advances in TCP friendly systems. We have presented the case through a simple 'interactive' generalization of the classical transport

[2] Interested viewer can retrieve both versions of the transported video from our website [4] for perceptual comparison.

control protocol, and a novel implementation of a symbiotic MPEG-2 transcoder. The proposed *principle of protocol interactivity* can enable fundamentally new solutions to many of today's hard to tackle problems. In this paper we have demonstrated the case of quality conformant congestion control for time-sensitive video traffic.
The approach exposed the overall advantage of network 'friendly' applications. However, it also departs significantly from the mainstream TCP friendly systems that have been suggested recently in two senses. First, it does not add any new major component in network software structure. It only expects some form of interactivity directly from the concerned network protocols as a general interface feature. Secondly, the applications do not have to be design-dependent on other auxiliary indirect probing tools or network utilities, nor it excludes their use when available. Some of the information measured by the auxiliary tools suggested by other approaches might be already available (or are being estimated/tracked) at lower layers anyway. At least this is the case with TCP congestion. The direct protocol interactivity we propose thus seems to be the logical path that can avoid potential duplication of efforts.
There are several current approaches based on some form of additional modules at systems/ or network layers. Unfortunately, many of them add enormous complexity at these layers and thus may be very difficult to make them practical. A detail survey on these can be found in [13].

Indeed in a striking contrast, one of the principal strength of the proposed interactivity is it's surprisingly simplicity. Nevertheless, this approach adds lesser but yet some cost in the network layer. The actual cost depends on the intensity of coupling. Designer of application symbiosis unit must be aware of the potential cost of tight coupling between handler and caller. However, as shown by the results-- with a prudent design the impact on the network level transfer rate (based on low layer measurement), if any, can be widely surpassed by the gain made at application layer. However, an interesting safeguard of this scheme is that a wrong design will only affect the application at fault and will have no effect on network.

The work has been supported by the current DARPA Research Grant F30602-99-1-0515.

References

[1] Wetherall, Guttag, Tennenhouse, "ANTS: A Tool kit for Building and Dynamically Deploying Network Protocols", IEEE OPENARCH'98, San Francisco, April 1998. Available at: http://www.tns.lcs.mit.edu/publications/openarch98.html

[2] Information Technology- Generic Coding of Moving Pictures and Associated Audio Information: Video, ISO/IEC International Standard 13818-2, June 1996.

[3] Jacobson, V, Congestion Avoidance and Control, Proc. SIGCOMM' 88, Conf, ACM, pp-214-329, 1988.

[4] Javed I. Khan, Q. Gu, Network Aware Symbiotic Video Transcoding for Instream Rate Adaptation on Interactive Transport Control, IEEE International Symposium on Network Computing and Applications, IEEE NCA' 2001, October 8-10, 2001, Cambridge, MA, pp.201-213

[5] Keesman, Gertjan, Hellinghuizen, Robert Hoeksema, Fokke Heideman, Geert, Transcoding of MPEG bitstreams Signal Processing: Image Communication, Volume: 8, Issue: 6, pp. 481-500, September 1996.

[6] Javed I. Khan, Seung Su Yang, Qiong Gu, Darsan Patel, Patrick Mail, Oleg Komogortsev, Wansik Oh, and Zhong Guo Resource Adaptive Netcentric Systems: A case Study with SONET- a Self-Organizing Network Embedded Transcoder, Proceedings of the ACM Multimedia 2001, October 2001, Ottawa, Canada, pp617-620

[7] Javed I. Khan, Qiong Gu and Raid Zaghal, Symbiotic Video Streaming by Transport Feedback based quality rate selection, Proceedings of the 12th IEEE International Packet Video Workshop 2002, Pittsburg, PA, April 2002, http://www.pv2002.org .

[8] Javed I. Khan, Darsan Patel, Wansik Oh, Seung-su Yang, Oleg Komogortsev, and Qiong Gu, Architectural Overview of Motion Vector Reuse Mechanism in MPEG-2 Transcoding, Technical Report TR2001-01-01, Kent State University.

[9] MPEG-2 Test Model 5, [URL: http://www.mpeg.org/MPEG/MSSG/tm5/, last retrieved December 25, 2000]

[10] Tennenhouse, D. L., J. Smith, D. Sincoskie, D. Wetherall & G. Minden., "A Survey of Active Network Research", IEEE Communications Magazine, Vol. 35, No. 1, Jan 97, pp 80-86.

[12] Dorgham Sisalem, Adam Wolisz, Towards TCP-Friendly Adaptive Multimedia Applications Based on RTP, Proceedings of the The Fourth IEEE Symposium on Computers and Communications, 1998.

[13] Javed I. Khan and Raid Y. Zaghal, Event Model and Application Programming Interface of TCP Interactive, Technical Report TR2003-02-03 February 2003, Technical Report TR2001-01-01, Kent State University, [available at URL http://medianet.kent.edu/ technicalreports.html, also mirrored at http://bristi.facnet.mcs.kent.edu/medianet]