A POLYMORPHIC FORMALISM FOR MADE-TO-ORDER CUSTOM CHANNEL BUILDING FOR LARGE SCALE NETCENTRIC SYSTEMS

JAVED I. KHAN

Media Communications and Networking Research Laboratory
Department of Math & Computer Science, Kent State University
233 MSB, Kent, OH 4424
330-672-9038 (o), 330-672-7824 (fax)
Javed@kent.edu

Abstract

In this paper we share our work and vision towards a radical new netcentric systems building formalism and associated mechanisms where custom soft composable channels can be built for fulfilling the varying but specific needs of the applications. These will also provide a novel bi-directional interaction mechanism between the application and network processes including event notification. The channels backed by the formal theory of polymorphic construction can prove to be a means for engineering a new generation of large, complex and smart network based systems.

1. INTRODUCTION

Internet applications are becoming increasingly complex. These are no longer only end-to-end systems where the Internet is a closed intermediate box and applications attach only at the end-points. It seems that any Internet centric application requires much higher level of embedded services entrenched inside the network, than what is available through the traditional and somewhat antiquated set of services offered in classical networking. An ominous manifestation of this fundamental need is the growing complexity of web-systems with proxies, caches and firewalls embedded deep inside the network. It symbolizes how a modern application requires storage, transformation, filtering, redirection, security and many other high level services. A future example of such system can be a virtual university with hundreds of sites and distributed campuses and management centers distributed globally, or a worldwide media distribution network, with global reach. A potential view of the future is that the network of computers will transcend to a networked system of systems. Future netcentric applications will be no less demanding in their need for similar yet-unheard-of entranced network services. Clearly, a far more advanced building technology will be needed for cost efficient construction, management, maintenance, expansion, and upgrade of such complex applications. Are we ready?

Low-level network architecture, particularly the performance has enormously advanced through rapid innovations in recent years. However, as far as applications are concerned, the only service model and the conduit to these advancements available to them to date is the TCP or UDP socket (and their minor variants) [1]. While they are proven tool for many traditional applications, there are also already many modern network applications for which neither seems to be quite suitable [2,3,4,5,6]. They require more sophisticated and customized communication service from the underlying network. If these are at all realizable, applications have to recompose them from ground up-- often reinventing wheels.

Should we build yet another 'perfect' service interface and accompanying network service layer to accommodate the newest needs of the emerging applications? It seems the fundamental problem is not with the TCP or UDP in particular— but any single such standard service stack will continue to lose cure-for-all appeal as applications are leaping forward to build enormously complex distributed systems.

It seems a better solution may lie in a radically new meta approach— *programmable channels*. We propose a particularly interesting formalism for the infusion of programmability into network where custom soft composable channels can be built for fulfilling the varying but specific needs of the applications—we call it *made-to-order channels* (MTO channels).

In this paper we share our work and vision towards a formalism and associated mechanisms that will enable building reusable and soft composable custom channels. The formalism allows building a new breed of sockets-- where channels also provide interactivity enabling dynamic exchange of local states between network and the applications. Potentially, not only a new class of applications but also a new class of solutions to many of the current hard-to-tackle network layer problems can be found [7,8,9].

2. BACKGROUND: WHERE ARE WE?

Before we present our idea/proposal, it will be interesting to have a brief tour of some of the exciting research in network centric applications and systems

area. Any network-based system is a composition of two types of elementary constructs-- the *process* and the *channel*. A number of research areas-- including parallel processing and middleware, have explored system building formalisms with distributed processing components (from PVM, to DCE, RPC [11,12], DCOM/COM, CORBA [13,15], Java/RMI) [15]. More recently, active networking [9,22] is exploring the means for adding programmability into network data path. Traditional network elements that have been deployed until now perform only a limited set of basic operations on the packets (such as forwarding and routing, fragmentation, packet dropping). Active Network proposes the generalization of the concept by incorporating almost unlimited programmable packet processing ability into the network elements. At least eight formalisms have been explored with great promise. Examples include SmartPacket [17], ANTS [9], PLAN [18], NetScript [19,22], and our Virtual Switch Machine [20,21]. If we look into the development of the current systems research (middle ware technology), we will see most of the previous formalisms for building complex networked systems have eventually focused on building more complex process construct and used standard elementary transport channels for connectivity even at the highest level of process abstractions. What is missing is the polymorphic abstraction of the channel construct. A software system construction framework imposes a set of *ontological* and *epistemological* constrained on the *components* or *process* and a set of *protocol* and *lexical* constraints on their *rules of interaction* [14]. A well-defined language for constraint specification also provides means to express properties flexible enough to be felt as programmable yet disciplined enough to ensure **inheritance** and **polymorphism.** These two properties are known to be the 'secret' for building complex system (even nature use them!). Iterative abstraction multiplies component reusability and enables iterative construction of large complex programmed systems in finite unique steps. Most of the recent research initiatives in middleware technology build on the impressive advances in component abstraction of Object Technology [15]. How do they stand on abstracting the rules of component interaction? Ironically all approaches [25, 26] focuses on providing ingenious domain specific variants-- but still non-polymorphic instances of channel constructs.

Given the advances the decade 2000's middleware technology has unveiled, the remaining question is simple-- *will it be possible to build a formalism that will enable a similar polymorphic abstraction of the channel construct*? And of course the ultimate query is-- *will it be possible to obtain a symbiotic pair of component and channel abstraction formalisms to work as a single polymorphic system providing the ultimate formal method (and ensuing technology) for building complex netcentric software systems?*

3. WHERE DO WE WANT TO GO?

The functionality, however is the most important issue (A null channel however meets both the above criterions). What new capabilities are needed into such polymorphic construction formalism? With the increased heterogeneity of network and attached devices [28], adaptation is rapidly becoming a crucial and integral part of the Internet based systems and applications [10,16]. Not only communicating systems will need new form of design time composition ability for building custom services optimized for their specific need, but they will also require run-time **self-organization, self-stability, adaptation** and the ability to generate custom response based on specialized events and varying operating conditions. The more globalization we demand from a system, it also means more powerful ability to localize. However, current formalisms do not provide any easy means for exchange of local state information between the network and the applications making it very difficult to build any adaptive or targeted solution system. What does it mean for channels?

From the mechanism/functionality point of view it requires that (ii) network local states critical for adaptation should be accessible to channel processes. (ii) A channel processes should have means to act locally inside network and reciprocate. (iii) The classical Internet channels only allow interaction between the subscriber (end-points) processes, where the underlying network is kept hidden by hard design. The provisioning of custom adaptation now requires that interaction have to be extended also between the subscriber (network layer processes) and infrastructure processes. (iv) Beside the data flow type of inter-process communication, support for other frequently encountered inter-process communication types/forms have to be built. In this paper, we particularly show a mechanism that allows building reusable and soft composable custom channels meeting many of the above criterions. The formalism allows building qualitatively a new breed of sockets-- where channels also provide interactivity and information exchange with end-points enabling dynamic exchange of local states between network processes and the subscriber processes. A *channel* means a coordinated set of communication services to a patron systems/application for sharing information between its two or more end-points. A channel itself generally has one source-end, one or more sink-end(s), and possibly a set of embedded network programs. The novel

channel that we envision [21] extends this classical notion in the following ways:

- **Programmability:** By programmability we refer to the service model where all the modules are programmable, not only the end-points. A third party channel designer can design and make it available to others. Installation and operation of the channels should be automated and should not require low-level administrative privileges so far it conforms to the design formalism and its security model. In addition patrons should be able to further configure its run time behavior by soft configuration without cumbersome recompilation/ reinstallation. It will be up to the channel designer to define the configuration options s/he wants to make available to the subscriber processes.

- **Event Notification:** Channels should have a mechanism to report designated events to patrons. A patron should be able to switch on or off notification features. When ON, the end-points should be able to receive interrupt by event handler routines. It will be up-to the channel designer to grant trap and trigger privileges.

- **Status Polling:** A subscriber system should also be able to read and write designated status variables in a channel. The channel program modules should have mechanism to query and poll local network states and receive notification of local network events. It should also be able to synthesis states before propagation. It will also be up to the channel designer to set the read/write privileges on these variables.

4. TECHNOLOGICAL BARRIERS

There lies significant multifaceted research challenge and technology barrier. This will require radical redesign of the switch architecture. Not only the network layer and above, but it will require additional services integrated with CPU scheduling, flow diversion, resource modeling, local I/O services from the switch OS. Where do you stand today? Switch today, only performs forwarding of cells and packets—here we are talking about radical generalization where switch should be able to run custom programmable modules. Traditional network systems have been designed mostly as a closed box system to applications. Even to upgrade a network embedded software, it requires physical administrative access to the routers. With programmability, not only bandwidth and input/out buffer (where most of our studies in high performance networking research are focused), even the computational cycle and processor memory becomes resources and starts a whole new avenue of research. At the end-points while the current socket/TCP mechanism has perfected itself through years, bi-directional interactivity between software components in an event driven process model in network condition is very little understood. Scores of new protocols have to be ironed out at network layer, for new services such as remote loading of processes, channel interception, security, programmable packet definition, channel relative addressing, system wide fault tolerance, etc.

Interestingly the potential pay-offs are equally high. It's expected that a composable channel when integrated into the design process of large and complex netcentric applications, will bring major benefits. Not only will it lead to substantial decreases in the development time and cost of their management, optimize their performance and ensure appropriate service model, but it can also jumpstart a new generation of otherwise hard-to-build applications which can take advantage of the interactive network local state exchange. The ability to access local network states can enable means for implementing a new generation of **adaptive network based applications**. It can also provide new techniques for attacking some notorious problems such as **congestion management** and **quality-of-service** provisioning. The ability to place application logic embedded into the network can empower a wide range of QoS applications ranging from MEMS based control, sensor fusion, co-ordination, scalable information filtering and search, distributed simulation, intrusion detection, to self stabilizing fault-tolerant systems.

5. EXPERIMENT & STRATEGY

We have outlined a three-part research-system-architecture for quick experimentation on building such a concept MTO channel capable network. Our critical path exploration pursuit targets (a) an interactive channel programming formalism through which applications will access and use the programmable channels, (b) the channel construction formalism for specification, construction and deployment, (c) and an active network inspired switch architecture and operating service model for supporting the building and execution of the programmable channels.

The base component is a **Virtual Switch Machine** (VSM) abstraction. The VSM is the embodiment of a modified switch OS and provides the stream interception (or switching) and routing facility over the switch fabric hardware and attached switchlet execution hardware (CPU, disk, memory). VSM provides three core services (i) it intercepts streams and inserts channel code modules (*switchlets*) in the flow

path, (ii) remote loading/unloading of the switchlets and (iii) resource management (resolves contention for e*xecution cycles*, *switchlet session memory*, and *forwarding order*. It allocated but does not actively resolve contention for ports, connected link bandwidth and local file space (they are not in the super critical path-- although it is possible to add such feature). See **Medianet Active Switch System** developed at Kent [20] for detail.

The application or channel programs are not intended to have raw access to VSM services and resources. Rather the code modules required to build the channels have to follow additional disciplines. The channel components (called capsule) are executed within a second abstraction layer called **Made-to-Order Channel Building & Execution Environment** (Channel BEE), which in addition to enforcing these restrictions—also provides a set of comfort utilities. BEE is composed of three service function sets— one of which is required by the patron programmer (EP Service), one by the transponders (CP Service), and the other by the BEE itself (NOS Service).

Finally, the channel. Description of each channel is called a channel **forma**. A forma has the codes (or URL pointers) for the **capsules** a **port table** describing the **connection matrix** connecting the capsules, and a **deployment map,** specifying the topological deployment plan for the modules. Each channel has **actuator** end-point, **audience** end-point and **transponder** capsules. For each instance of a channel generally there should be one instance of the actuator, one or more instance of audience(s), and zero or more instances of transponders. The details of this work can be found in [20,21]. In the next section we will share the most interesting part—some example MTO channels that we are building on the VSM/BEE.

6. EAMPLE MTO CHANNELS

To illustrate how the MTO channel formalism can spark fundamentally new solutions to many hard problems and facilitate complex netcentric systems and applications building we will illustrate few MTO channels that we are working one.

6.1. Daisy Chain Replenishment Channel [21]

This (DCR) channel is intended for carrying massive video feed files from the *Origin Servers* to the strategically located *distribution cache servers* via long haul shared network utilizing mostly the off-peak hours of the involved hops. In a long haul network all the hops may not have the off-peak simultaneously. The idea is to install a series of *transfer cache* stops at appropriate network points. If a link is busy, transfer cache automatically stores the transit traffic in a secondary buffer (such as hard drive). When the network load eases, it then forward the waiting data to the next stop. As evident, synchronization is dependent on local network state, but the application end-points are relieved from worrying about the details of communication synchronization. End-points simply subscribe and install the DCR-channel, specify a delivery schedule (such as “deliver within 3 hours”) and return to their original work, while the MTO channel handles the details of the trucking operation. The channel has additional smart interactive status and event services. For example, when a deadline failure is anticipated based on local network ‘weather’ conditions in the flight path, it can send notification to the receiver application. Also channel can accept ' sweep' signal. A patron may change its mind (such as an user has requested a video prior which is still in transit) and requests all pending data to be delivered immediately. An iteratively enhanced version of this channel -- the **DCR-parallel** extends it for parallel communication by including a disjunctive parallel path discovery mechanism between the transfer caches for higher performance and increased reliable by concurrent communication.

6.2. Rate Adaptive Transcoding Channel [23]

We have also experimented with another novel MTO channel for carrying live video feed via network with highly asymmetric resources. It offers adaptation at two levels. First, the channel has the capacity to detect local congestions at any point inside a network. If it detects congestion it is capable of dynamically re-encoding the video at a lower rate to adapt with the variations in link bandwidth. When the congestion eases it is able to resume its original stream rate. The channel has been implemented using embedded transcoder capsules. In the second level, it offers adaptation based on variations in the processing capability on the switches. An auxiliary set of capsules exchanges very low impact background communication to track the dynamic frame rates for **self-reorganization** into low computation but relatively higher distortion based transcoding. Its’ interactive features includes notification to the application-end-points when the rate conversion kicks in or exceeds a certain limit. The channel can splice **wireless network** with the fiber Internet or bridge the **digital divide**. A design of this nomadic self-organizing system with multilevel adaptation ability has been outlined in out just accepted publication [23]. Based on this novel formalism the core idea can almost routinely be expanded to demonstrate **self-assembly, self-configuration, and self-repair** and other forms of self-organizing behavior.

6.3. TCP Interactive:

Recently, we have also simulated an **interactive version of the classical TCP** channel (called TCP interactive), which has few network event notification mechanisms to the end points. If there is a congestion event (such as change in TCP window size, packet discard), we demonstrate a novel channel, which also notifies the application. Where the augmented with smart application modules, we were able to demonstrate dramatic improvement in video QoS conformant MPEG-2 communication. For example, where a classic TCP suffering 50% packet loss, we were able to demonstrate 100% frame delivery with the same network resource, where the catastrophic frame delay was traded off for acceptable reduction in SNR quality, resulting in revolutionary advance in state-of-the-art. More details of this interesting scheme can be found in [24].

6.4. Time Routed Channel:

One of the critical class of applications which cannot be conveniently supported in current end-to-end network infrastructure is the time bounded applications particularly deadline based systems' communication and control systems with time sensitive feedback loop. One of the critical steps in the pathway of all such communications is the routing. Unfortunately, current *first-come-first-served* (FCFS) Internet forwarding does not allow for time bound communication. We have also performed some experiment to study the potential architectures for facilitating time critical communication with the MTO channel formalism. The scheme does not require a built-in time based routing in the core switch level (VSM). Rather some form of priority based differential forwarding is sufficient. The transponder capsules can perform the actual time-based priority assignment [7].

Indeed more than 30 concepts of useful MTO channels have emerged (few have attracted proprietary interest), each of which can enormously simplify the communication substructure of network applications. Overall a well-endowed library of channels can radically accelerate the ability to build and maintain the emerging netcentric application of unprecedented complexity by streamlining their communication synchronization sub-structures. Almost 30-40% of code complexity of some existing network based applications already lies here.

The concept of MTO channel in essence creates a powerful customizable and adaptable middle layer mechanism for close loop interaction between the information and network components for real-time coordination and synthesis. The MTO channel formalism can be customized for almost any closed loop interactive systems for real-time synthesis and coordination. They can be used for wide applications ranging from MEMS based control, sensor fusion, co-ordination, scalable information filtering, information search, distributed simulation, intrusion detection, to self stabilizing fault-tolerant systems.

7. POLYMORPHIC MTO CONSTRUCTION

The real power of the proposed MTO channels lies in an iterative building process, which can lead the way of building enormously complex netcentric applications. In this mechanism in the first stage, using a channel composition formalism $\Im_{channel}$, a set of made-to-order (MTO) channels are built by VSM/BEE using a set of elementary capsules and a set of elementary channels. In the second stage using the capsule composition formalism $\Im_{capsule}$, a newer and more complex set of capsules can then be composed. In a recursive mode, in further iterations, the MTO channels of the previous stages can be used as the building blocks for composition more complex MTO channels and capsules. The power of this iterative formalism is derived from **inclusion polymorphism** of the MTO channels. Once built, they can be used in all subsequent higher order construction stages due to their self-similarity. Each aggregation also seems to be capable of **inheriting** the associated event set without violating their component property. It can be shown that today's network protocol service layered architecture is merely a special instance of this powerful formalism. The IP layer (including the routing, ARP and auxiliary protocols and the IP end-points) is the highest channel construct with all three types. The service layers above IP including TCP are special cases with null transponder sets. Furthermore, any client-server application at the top can be viewed as the highest order capsule construct where the server is a module of type actuator and the client of type audience, with null transponder. Further understanding of the formalism pair $\Im_{capsule}$ and $\Im_{channel}$ hold the key to many issues involved in the development of complex netcentric applications.

8. BROADER IMPACT

The classical socket/TCP/UDP bundle has served us reliably and for a long time for many classic applications such as email. But they are showing the sign of age as new generations of complex applications are burgeoning. TCP or UDP? The current literature is abundant with studies, which argues inconvenience of one over the other for video (or QoS sensitive data in general). Are we stuck between the two?

There have been enormous innovations in the layers beneath. But, it seems without any fundamental change in this interface the end-impact that eventually reaches the subscriber processes reduces to something only quantitative in nature. The proposed construct of *interactive* and *programmable* channel is expected to bring fundamental qualitative advantage to the netcentric systems. It can also enable new optimization techniques for attacking some notorious network problems such as *quality-of-service* (QoS) provisioning. As far as applications are concerned, it seems the issue of QoS is merely not a numeric one. Indeed much of the problem arises from the definition and composition of what do we mean by 'service' itself. Programmable socket seems to be the right first-step in this provisioning.

Conceptually, the proposed formalism resembles closely the enormously successful easy to perceptualize abstraction of the classic 'communication channel'. However, we include radical generalization in its functionality, which can revolutionalize future netcentric applications— yet it will remain intuitive as its classical counter part and will offer full backward compatibility. The concept of MTO channel in essence creates a powerful customizable and adaptable middle layer mechanism for close loop interaction between the smart subscriber applications and network components with real-time coordination and synthesis. Not only it can empower communicating systems with a new form of design time composition ability for building custom services optimized for their specific need, but the interactive capability will provide run-time adaptation and the ability to generate custom response based on specialized events and varying operating conditions inside network.

Clearly there lies significant multifaceted technological challenge. This will require radical redesign of the switch architecture. Scores of new protocols have to be ironed out. It took more than two decades to perfect the classical socket formalism. The proposed attempt to make this classical socket programmable as well as interactive is a massive escalation in system's complexity by this measure. The proposition is surely difficult and challenging but definitely not impossible. The recent initiative such as active network indicates that insertion of such programmable component might be within the reach if resources are allocated. The challenge however, is clearly worth exploring. The recent emergence of the Internet now opens up the ground for building network based software systems with unprecedented complexity. This will however need rapid qualitative innovation in current networking technology.

Network bandwidth is routinely doubling in an astonishing rate of every 9 months [27]. Bandwidth alone will not be able to meet the challenge of 'digital convergence' [28]. At the least, such quantitative gain has to be translated into qualitative advancement if such applications are to be engineered.

Some of the work cited here has evolved from the ongoing DARPA/ITO project PERCEPTMEDIA (Grant F30602-99-1-0515).

9. Reference

[1] Stevens, W. R., UNIX Networking Programming, Networking APIs: Sockets and XTI, v.1, 2nd ed, Prentice hall, PTR, 1998, NJ.

[2] Javed I. Khan, Prefetch Scheduling for Composite Hypermedia, IEEE International Conference on Communications, ICC2001, June 2001, Helsinki Finland (accepted as full paper)

[3] E. Amir et al. An Active Service Framework and its Application to Real-time Multimedia Transcoding. In *Proc. SIGCOMM'9*8, pages 178–189. ACM, Sep 1998.

[4] E. Johnson. A Protocol for Network Level Caching. M.Eng Thesis, MIT, May 1998.

[5] C. Papadopoulos et al. An Error Control Scheme for Large-Scale Multicast Applications. In Conf. on Computer Communications, INFOCOM'98, San Francisco, CA, April 1998 [IEEE Digital Library URL http://computer.org/publications /dlib/]

[6] Javed I. Khan and Qingping Tao, Proxy Cache Integrated Partial Prefetch for Faster Surfing in Composite Hypermedia, 3rd USENIX Symposium on Internet Technologies and systems USITS ' 01 March 26-28, 2001, San Francisco, USA, (accepted as full paper).

[7] Javed I. Khan, & Numan Bantan, Performance of Queue Scheduling Algorithms for Time based Routing, Technical Report: 2000-11-01, Kent State University, [available at URL http://medianet.kent. edu/ technicalreports.html, also mirrored at http:// bristi.facnet.mcs.kent.edu/medianet]

[8] K. Ramakrishnan and S. Floyd. A Proposal to add Explicit Congestion Notification (ECN) to IP. RFC 2481, IETF, Jan. 1999. [Available at URL ftp://ds.internic.net/rfc/]

[9] Wetherall, David, Active Network Vision and Reality: Lessons from capsule-based System, Operating Systems Review, 34(5): pages 64-79, December 1999.

[10] Gabor Karsai, Janos Sztipanovits, A Model-Based Approach to Self-Adaptive Software, IEEE Intelligent Systems, Vol. 14, No. 3: MAY/JUNE 1999, pp. 46-53

[11] Coulouris, G, and Dollimore, J, Distributed Systems: concepts and Design, Addison Wesley, 2nd ed., 1994.

[12] Corbin, J. R., The Art of Distributed Applications, Programming Techniques and Remote procedure Calls, Springler-Verlag, 1991.

[13] D. Schmidt, D. Levine, and C. Cleeland, "Architectures and Patterns for High-Performance, Real-Time CORBA Object Request Brokers," Advances in Computers, M. Zelkowitz, ed., Academic Press, 1999.

[14] Lee, Edward, What's Ahead for Embedded Software, IEEE Computers, Vol. 33, No. 9, September 2000, pp-18-26.

[15] Gopalan Suresh Raj, A Detailed Comparison of CORBA, DCOM and Java/RMI, [URL: http://www.execpc.com /~gopalan/ misc/compare.html,retrieved on Dec. 1999]

[16] Robert Laddaga, Creating Robust Software through Self-Adaptation, IEEE Intelligent Systems, Vol. 14, No. 3: MAY/JUNE 1999, pp. 26-29

[17] Beverly Schwartz, Wenyi Zhou, Alden W. Jackson, W. Timothy Strayer, Dennis Rockwell, Smart Packets for Active Networks. In 2nd Conf. on Open Architectures and Net-work Programming, OPENARCH'99,NY, Mar. 1999. [URL: http://www.ir.bbn.com/~bschwart/, Last retrieved 10//02/00]

[18] M. Hicks et al. PLANnet: An Active Internetwork. In Conf. on Computer Communications, INFOCOM'99, pages 1124–1133, New York, NY, Mar. 1999. IEEE.

[19] Y. Yemini and S. da Silva. Towards Programmable Networks. In Intl. Work. on Dist. Systems Operations and Management, Italy, Oct. 1996. [URL: http://www.cs. columbia.edu/dcc/netscript/Publications/publications.html, Last retrieved: 11/02/00]

[20] Javed I. Khan, S. S. Yang, Medianet Active Switch Architecture, Technical Report: 2000-01-02, Kent State University, [available at URL http://medianet.kent. edu/ technicalreports.html, also mirrored at http:// bristi.facnet.mcs.kent.edu/medianet]

[21] Javed I. Khan & S. S. Yang, Made-to-order Custom Channel for Netcentric Applications over Active Network, Proc. of International Conference on Internet and Multimedia Systems and Applications, IMSA 2000, November 20-23, 2000 Las Vegas, USA, pp22-26.

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

[23] Javed I. Khan & S. S. Yang, Resource Adaptive Nomadic Transcoding on Active Network, International Conference of Applied Informatics, AI 2001, February 19-22, 2001, Insbruck, Austria, [available at URL http://medianet.kent. edu/ technicalreports.html, also mirrored at http:// bristi.facnet.mcs.kent.edu/medianet] (accepted as full paper).

[24] Javed I. Khan & Q. Gu, Application Integrated Congestion Control by made-to-order Channel Composition on Active Network, International Conference of Applied Informatics, AI 2001, February 19-22, 2001, Insbruck, Austria, [available at URL http://medianet.kent. edu/ technicalreports.html, also mirrored at http:// bristi.facnet.mcs.kent.edu/medianet] (accepted as full paper).

[25] Donatella Sciuto, IEEE DESIGN & TEST OF COMPUTERS, Vol. 17, No. 2: APRIL-JUNE 2000, pp. 11-13, Guest Editor' s Introduction: Design Tools for Embedded Systems.

[26] Matti A. Hiltunen, Richard D. Schlichting,, Xiaonan Han, Melvin M. Cardozo, Rajsekhar Das,Real-Time Dependable Channels: Customizing QoS Attributes for Distributed Systems, IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS,Vol. 10, No. 6: JUNE 1999, pp. 600-612

[27] Gary Stix, The Triumph of the Light, Scientific American, January 2001, pp31-35 [HTTP:// http://www.sciam.com/2001/0101issue/0101stix.html]

[28] [Peter Forman and Robert W. Saint, Creating Convergence, Scientific American, November 2000, pp.10-15[HTTP://http://www.sciam.com/2001/0101issue/0101stix.html]