Practical Active Network Services within Content-aware Gateways

Siva Subramanian[1] , Phil Wang, Ramesh Durairaj, Jennifer Rasimas, Franco Travostino
{ssiva, pywang, radurai, jrasimas, travos}@nortelnetworks.com
Advanced Technology Investments, Nortel Networks
Tal Lavian
tlavian@nortelnetworks.com
Computer Science Division, University of California, Berkeley
Doan Hoang
dhoang@it.uts.edu.au
Department of Computer Systems, University of Technology, Sydney

Abstract *The Internet has seen an increase in complexity due to the introduction of new types of networking devices and services, particularly at points of discontinuity known as network edges. As the networking industry continues to add revenue generating services at network edges, there is an increasing need to provide a systematic method for dynamically introducing and providing these new services in lieu of the ad-hoc approach that is in use today. To this end we support a phased approach to "activating" the Internet and suggest that there exists an immediate need for realizing Active Networks concepts at the network edges. In this context, we present our efforts towards the development of a Content-aware Active Gateway (CAG) architecture. With the help of two practical services running on our initial prototype, built from commercial networking devices, we give a qualitative and quantitative view of the CAG potential.*

Keywords: *Active networks, Programmable networking devices, Content Delivery Networks, Content Networking, Content Transformation, Streaming media, Multicast, Network-edges, Gateways.*

1.0 Introduction

The advent of the world wide web and commercialization of the Internet has led to an explosion of content providers and users. The Internet faces challenges arising from economic factors and legal issues in the telecom market, deficiencies in the existing network infrastructure and increased demand for bandwidth and services. This has resulted in the creation of a new concept: Content Networking [25-27]. Content networking overcomes the inadequacies of existing networks by introducing intelligence into the network in order to enhance performance of services and delivery of content to the consumer. It has become important primarily due to the potential for generating new revenue from such intelligent services [28-30].

Content Delivery Networks (CDN) are typically implemented as overlay networks and contain one or more nodes that can inspect and/or manipulate information in higher networking layers (four through seven in the OSI reference model). Examples of CDNs include web cache networks and video streaming networks. Examples of content networking devices include voice packet gateways, web caches, load-balancing switches and firewalls [28-30]. Content Networking services are typically introduced into the network in the form of dedicated custom networking devices. Such an approach exhibits increased complexity and redundancy and also poor manageability and upgradeability.

1. Siva Subramanian is a Ph.D. candidate in the Electrical and Computer Engineering Department at North Carolina State University, Raleigh

Content networking devices are used at the edges of networks. The edge of a network indicates a demarcation of network types as shown in Figure 1, and is typically characterized by an impedance mismatch. The mismatch is typically due to differences in physical network attributes such as network size, link bandwidth, network latency, network capacity or abstract attributes such as trust, authority. Some examples of real-life impedance mismatches are listed below.

- low-speed access networks versus high-speed metro optical networks
- trusted enterprise internal network versus untrusted external network
- electronic networks versus photonic networks
- QoS-enabled networks versus best-effort networks
- secure encrypted network versus unsecured network
- wireless access network versus the wireline network
- circuit-switched network versus packet-switched network

Such discrepancies in the network boundaries can be overcome by additional processing of the traffic flowing across them. However, this additional processing is often not the same for all traffic types, and is typically not known a priori. Content Networking applications adapt the processing to the type of the traffic flowing across a given network edge. For instance, firewalls process traffic between trusted and untrusted domains by filtering and taking actions based on the type of content.

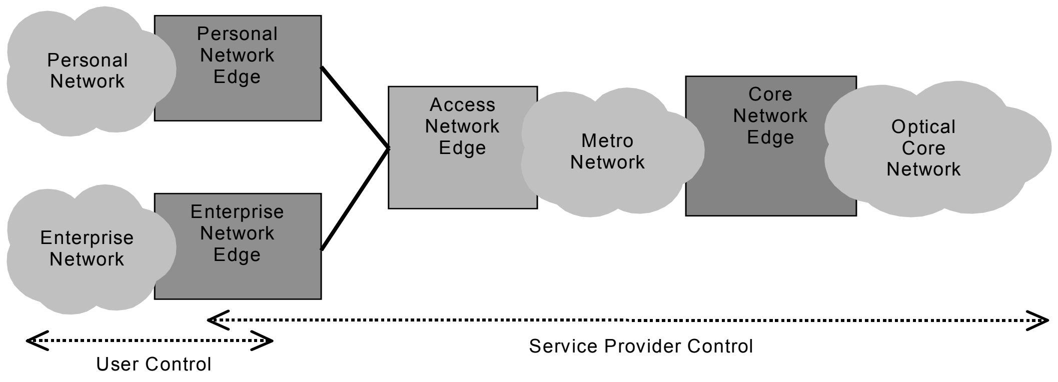

Figure 1: Network model with Edges

In this paper, we present a new application for active networks research concepts in the form of a Content-aware Active Gateway (CAG). The hardware and software architectures that we have developed are presented in section two. We have implemented a CAG protoype using real-world high-performance networking devices augmented with software that we developed. Details of the prototype are described in section 3. We have implemented two simple commercial services on the CAG prototype. These services and the results from our experiments are described in section 4. Section 5 presents some observations made during our experiments with the CAG prototype. We discuss related works in active networks in section 6 and finally present our conclusions in section 7.

2.0 Content-aware Active Gateway

We propose a new type of active networking device, the Content-aware Active Gateway that can be used to deploy a wide range of services at network edges highlighted in Figure 1. The services deployed on the gateways are controlled by the service provider or even the user as in case of a residential network edge. Contractual agreements between one or more service providers and users are part of the business model. The services are developed by service providers using software technology such as Java and can be deployed and enabled on-demand on the CAG. In an access network model where the CAG is owned by the network provider, the service provider leases processing capacity on the CAG in order to serve the users. In the telecom network model, a CAG can support services required for peering networks. In a residential service provider model it is possible that the user, under authorization of the service provider, may have control over the services that are "loaded" onto the CAG. The services can be either statically configured on the CAG or alternatively loaded on demand. Examples of dynamic services include content personalization, firewalling, virus scanning, encryption, compression and video chat servers. The ability to deploy services on demand into the network can lead to renewed activity in the service provider industry and new revenue sources for broadband network infrastructure providers. The process of introducing a new service can be simplified and limited to developing the software application and putting in place the contractual agreements between the gateway owner and the service owner. Eliminating the need for large investments needed to introduce new equipment into an existing network can lead to new ventures in the service development industry.

2.1 CAG Hardware Architecture

The main feature of a content networking device is the ability to filter specific types of traffic by identifying patterns in the header or payload of packets. The redirection capability of such a device enables processing of the filtered and redirected traffic. Thus the main components of the CAG Hardware Architecture are a fast packet forwarding and classification plane, a high-performance compute plane and a flexible control plane. The difference between a router and the CAG node lies in its distributed processing architecture comprising of multiple compute resources. The processing capability leads to new performance metrics such as Compute Quality of Service (CQoS), which, similar to network Quality of Service, can lead to service differentiation and therefore revenue potential. Routing and forwarding performance of a CAG, especially for traffic not requiring any transformation, must not be affected by the services executing on the compute resources. This requirement is similar to that of Active Routers [9,15,19] and can be achieved by decoupling the network processing resources at layers two and three from the application processing at higher layers.

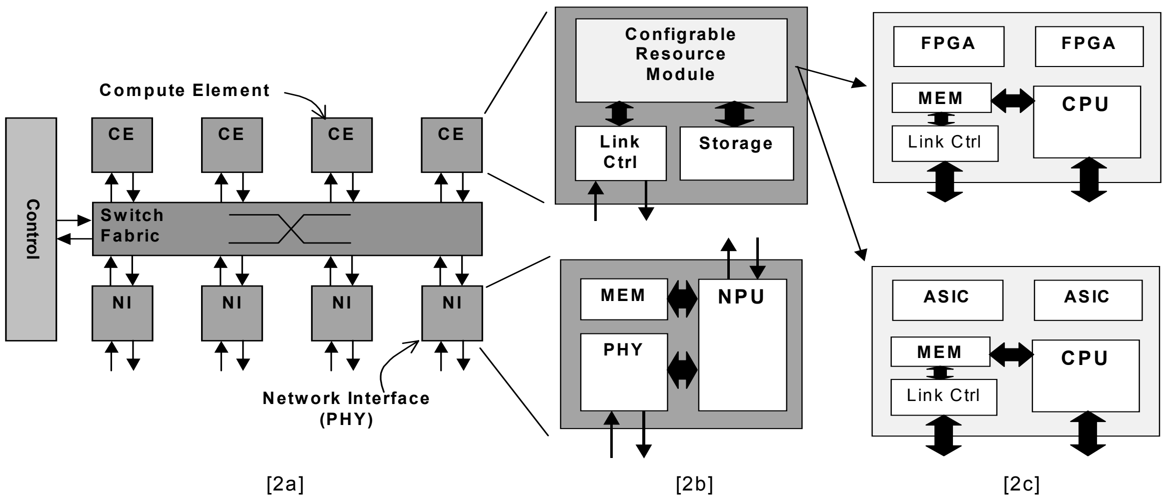

Figure 2: CAG Hardware Architecture:
[2a] Hardware Block Diagram,
[2b] Compute Engine (CE) & Network Interface (NI),
[2c] Configurable Resource Module (CRM) "flavors"

Compute requirements for the resources on a CAG node can vary depending on the application. These functions can be implemented over a range of technologies such as CPUs, NPUs, FPGAs and ASICs to achieve different levels of performance. A service demanding high CQoS can be allocated an FPGA based compute resource whereas the same service being offered by another provider at a lower CQoS (and therefore cost) can be performed on a general purpose CPU resource. The CAG architecture must allow a mix of these technologies to achieve the desired cost, performance and flexibility targets. The physical port resource as well as the compute resource on a CAG, is a shared resource enabling traffic from any PHY to be processed by any compute resource. This eliminates the need to duplicate services on multiple resources unless demanded by increased processing requirements. The modular architecture of the CAG node, shown in Figure 2a, consists of Network Interface (NI) modules, Compute Engine (CE) modules, the Switch Fabric and a Control module. Typically, these modules will be implemented in the form of blades or cards that sit in a single network equipment shelf connected by a backplane. Backplane connectivity between the modules will use point-to-point high-speed serial links for increased performance and reliability. Alternately the CE modules can be implemented by external hardware if physical port costs on the switching shelf are high.

The Network Interface module, as shown in Figure 2b, consists of one or more Network Processing Units (NPU), physical layer ASIC (PHY), packet memory and any additional silicon to perform the functions of filtering, classification and queuing. The NI module is capable of making all decisions in the layer two and three domain (switching, forwarding and routing). The CE module, shown in Figure 2b, executes the services downloaded to the CAG node. The CE module contains the Configurable Resource Module (CRM), link interface controller and also some permanent storage. This permanent storage can be flash memory or miniature hard disk storage and is used to store long-term state for the service. A mandatory component of the Configurable Resource Module (CRM) is the CPU. The CRM can employ one or more high density FPGAs and ASICs along with a CPU as shown in Figure 2c. FPGAs and ASICs on the CRM assist in processing of compute-intensive functions that cannot be efficiently handled by a CPU. Special purpose CRMs can also have connections to external peripherals, such as a RAID disk array, for a web cache. The switch fabric module uses one or more non-blocking high-speed crossbar silicon switching chipsets. The Control module performs the Operations Administration and Maintenance (OA&M) functions on the CAG.

2.2 CAG Software Architecture

Extensive research has been done on Active/Programmable Network APIs [2-5,7,10-12,16]. The features of such APIs include service creation and deployment, security and policy controls. The high-level software architecture of the CAG is shown in Figure 3.

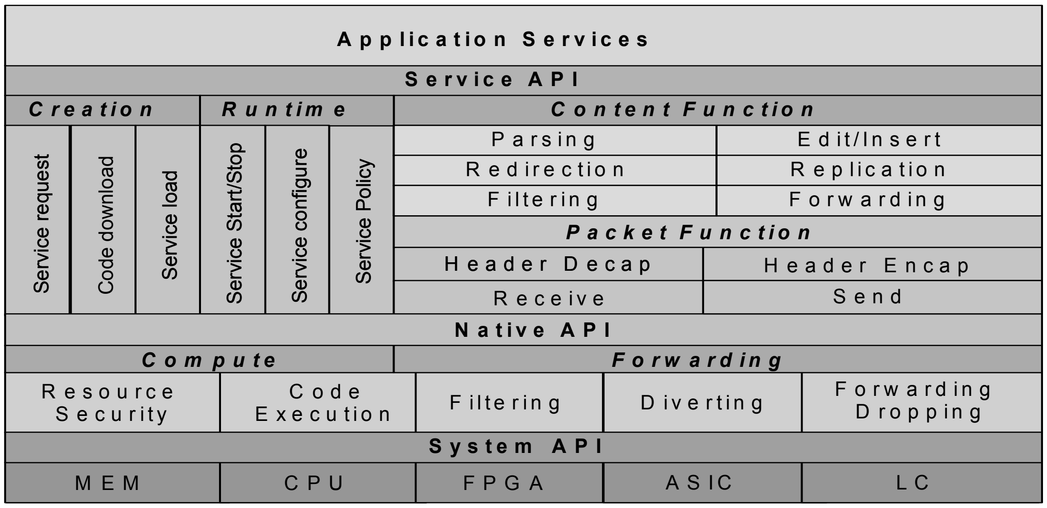

Figure 3: CAG Software Architecture

In the architecture, application services at the top are built by high-level Service APIs. These Service APIs provide applications the capabilities to create new services, execute existing services and perform content-based functionality. Service Creation API is required for service creation, installation and security. Service

Runtime API is required for computing or network resource reservation and allocation, configuration, signaling and management. The Content Function APIs provide the services to do various content manipulation, including content filtering, redirection, parsing, editing, insertion, replicating/multicasting and forwarding. Below the Content Function API is the Packet Function API which receives and decapsulates individual incoming packets, encapsulates and sends outgoing packets, and sets up any-layer packet filters, packet data buffer allocation, data capture, data insertion and tagging. Native API provides low-level access to native resources of computing and forwarding.

The Forwarding API supports the regular networking functions such as packet filtering, diverting and forwarding/dropping, typically realized in hardware. System APIs provide access to hardware or micro-code. The Compute API provides the functions required to configure the resources, read and write to memory or storage and resource-specific functions etc. These APIs are wrapped by the Configurable Content Networking API thereby hiding the complexity of the resource modules from the services. The Service APIs for service creation and runtime APIs are mainly inherited from the Openet SDK (ODK) [12, 15]. The ODK, is readily available from [28] and can be used to create and deploy Active Networks services onto commercial network devices.

3.0 CAG Prototype

The first prototype of the CAG was developed using existing commercial high-performance networking devices. The Nortel Alteon Webswitch [28], referred to here as the Alteon switch, is a commercial-grade content networking switch. It has line-speed filtering capacity on its Gigabit Ethernet ports and the ability to redirect traffic to any of the ports. This redirection feature allows for additional processing of traffic on an external device before being forwarded to the final destination. The switching fabric in the CAG architecture is implemented by the Alteon switch. The CE modules in our prototype are external devices based on 1GHz Pentium III CPU servers running Linux. One of the CE module uses an FPGA-based Compute Resource Module plugin. The FPGA-based CRM uses is a Xilinx Virtex 300E on a daughter card and plugs into the CE module. The CE modules connect to the Alteon switching fabric via Gigabit Ethernet links. We augmented the Alteon switch and the CRM with software modules that we developed, enabling it to behave as an active networking device. The network setup for our experiments is shown in Figure 4. The setup consists of two CAG node prototypes positioned at the two edges of a LAN. The right edge of the network in the illustration is the content provider edge and the left edge is the user edge. The API software we developed on the CRM can configure the Alteon switch to set filters, actions and also supports extraction and insertion of packets to and from the data path.

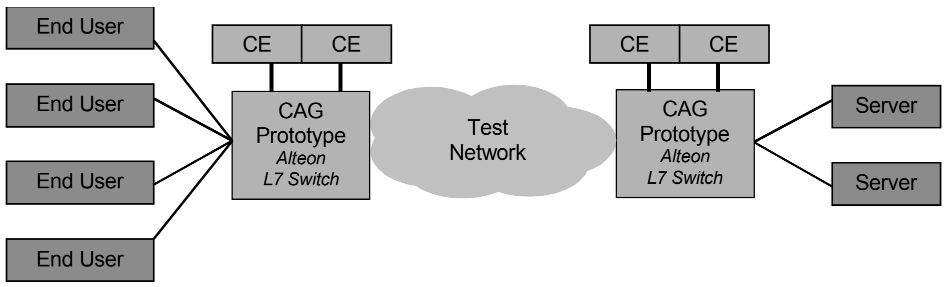

Figure 4: CAG Prototype Testbed

4.0 CAG Applications

In this section we present two practical content networking applications implemented on the CAG prototype testbed. The first is a streaming media service that can be used for webcasts, large-scale videoconferences, high-quality videoconferences etc. The second is a web service that transforms web content for delivery to end systems.

4.1 Streaming Media Distribution Service

Streaming media is becoming the de facto global media broadcasting and distribution standard, incorporating other media, including radio, television, and film [25]. IP multicasting allows multiple users of a network to listen to a single live stream, making efficient use of network resources. Multicasting avoids delivering numerous connections by broadcasting one stream to a certain point in the network where other users are requesting the same file. IP multicast would have been an ideal technique to broadcast data stream from a source to multiple destinations if it wasn't for several critical factors. First, the business model does not adequately cover the cost of replication of data at each intermediate router. ISPs costs for peering are roughly $700-$1,000 per Mb/s per month leading to an annual investment of about $1M for peering at 100Mb/s. There is no incentive for Internet Service Providers (ISPs) to deploy the multicast service model as it will not add to the revenues. Second, multicasting requires support inside the network in terms of elaborate control support from IP routers, membership managements (IGMP), and multicast routing protocols. Third, enterprises do not want to run multicast for fear of degrading the performance of other mission critical applications. Fourth, WAN connections are extremely expensive. Enterprises are not willing to pay the additional charges incurred from content streaming.

Let us consider a streaming video seminar application where "viewers" attend the e-seminar from multiple sites of an enterprise on the network shown in the Figure 5. The live video information originates from one of the sites and is delivered to several clients at different sites. In such a scenario, the access links that connect the edges of the enterprise network sites to the service provider network are typically the bottlenecks and tend to get congested due to the number of viewers and other interfering traffic such as web, email, etc. This is a problem when the number of viewers is large and each viewer establishes a separate connection to the source of the video stream, i.e. when the source has to multicast the video information. Although it is possible to design a custom solution to this problem, it only makes the network more complex as special purpose boxes are introduced to solve specific problems. On the other hand we can assume that the enterprise has deployed CAG nodes at the edges of its network. As described in this section, the use of the CAG edge nodes will allow for a manageable solution to the problem along with cost savings.

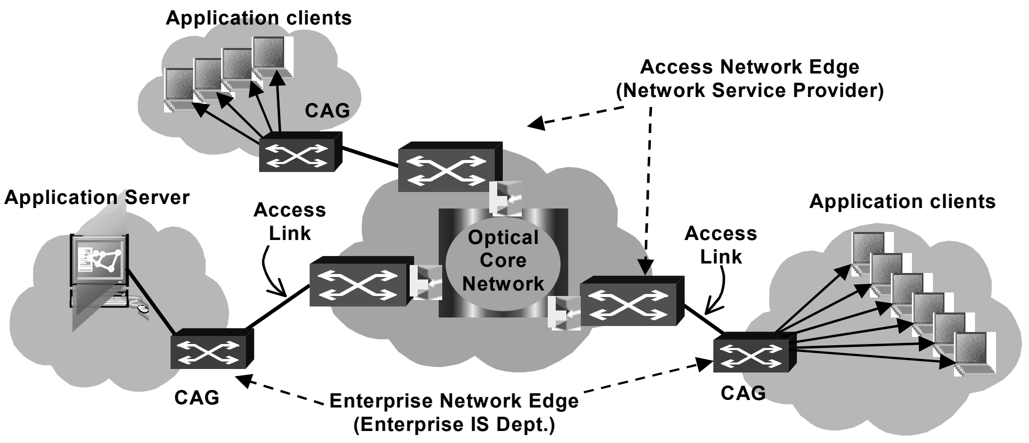

Figure 5: Streaming Video in a Distributed Enterprise Network

We introduce a service on the CAG, the Streaming Media Distribution Service (SMDS), that can achieve the effect of multicasting, but without having to upgrade the networks to support IP multicast or without having the application perform any multicasting across the access link [26]. The CAG nodes at the edge of each of the enterprise sites form a logical overlay network. The SMDS on a network architecture employing CAG nodes can provide high-quality streaming without access-link bottlenecks, and is scalable in terms of serving a large number of clients. We considered other efforts at solving the multicast problem. The CMU effort [18] tries to move the complexity of managing multiple streams to the application at the end system. However, this solution would still suffer from the access link bottlenecks. Other efforts such as

REUNITE [14] and HOP-by-HOP methods [20] require upgrading of the core infrastructure which is beyond the control of the enterprise.

The source video application server sends a single stream to the CAG node at the edge of its enterprise network. The SMDS on the source CAG edge node replicates and unicasts the source video stream to the destination CAG edge device at each of the client sites. Thus, the application does not have to perform any multicasting and the bottleneck problem at the source access link is avoided. A cluster of clients is always associated with a destination edge CAG node. Only one copy of the stream needs to be sent across the overlay network from the source edge to a destination edge. The SMDS on the destination CAG edge node has to simply replicate and unicast the received data stream to the clients in its cluster. The SMDS "relocates" the locus of video distribution to the edges of the client sites, thereby eliminating the bottleneck at the destination access links.

We developed and tested the SMDS on our CAG prototype testbed shown in Figure 4. The video source server used was the Real Server 8 on a Windows NT based system. The clients were off-the-shelf PC end systems. We deployed the SMDS on only one destination CAG edge node in our experimental enterprise network while the CAG node at the source edge simply acted as an IP router. The Real video system in our setup uses port number 554 for the RTSP protocol. The SMDS software makes use of this port number as well as the IP address of the Real server to detect the streaming media flow.

4.1.1 Operation Details

The SMDS service on the destination CAG operates as follows:

1. Once the SMDS is downloaded and enabled on the destination CAG, it sets up two filters on the Alteon switch via API calls. One filter is to trap any packets with the Destination IP address of the Real server on Destination port 554 and redirect the traffic to the SMDS on the CRM. The other filter is to trap all the content packets from the Source IP address of the Real server and Source port 554 and redirect them to the SMDS on the CRM.
2. The SMDS software application then binds itself to the end of each of these tunnels (resulting from each filter) and suspends waiting for the filters to trigger.
3. When the first client attempts to connect to the Real server, the first filter is hit and its request packets arrive at the SMDS. The SMDS registers the client into a local database using API calls.
4. The SMDS software then forwards the client connection requests on to the Real server and awaits the response.
5. When the Real server starts streaming video content back to the client, the second filter on the CAG is triggered and the content packets are redirected to the SMDS.
6. The SMDS software simply forwards the traffic onto the client that originally requested the content.
7. On successive connection requests from other clients for the same content as in step 3, the SMDS intercepts the request packets but does not forward them to the Real server.
8. The SMDS completes handshaking with each requesting client (acting as a proxy for the Real server) to complete the connection.
9. The SMDS on the CRM then starts duplicating the video content that is being sent by the Real server (to the first client) and forwards the duplicated traffic to each of the requesting clients.
10. On successful completion of the streaming media content, SMDS closes all the client and server connections and waits for a new request.

The core of the SMDS software, that generates the duplicate packets, (as in step 9) involves creating a new header with the appropriate destination IP address and port (of the client), filling the packet with payload from the original content packet and computing the new IP checksum before sending the packet to the physical link.

4.1.2 Performance Measurements

We conducted a simple experiment to evaluate the performance of the SMDS on our prototype. Measurements were taken at the client nodes using tcpdump and tcptrace to obtain the throughput. We used a source file for the video streams encoded at 1.5Mbps. We used four client PCs running RealPlayer software to view the streaming media. We measured performance in two scenarios: first using regular unicast streams and the second using SMDS. The results of the measurements are shown below in Table 1.

The measurements indicate that the SMDS service eliminates the problem of unfair distribution of available bandwidth which is seen when using multiple unicast streams directly from the server. The unfairness is partly due to TCP behavior and partly due to the Real Media server. Also the bottleneck with the unicast scenario is clearly at the access link which was measured at about 4.3 Mbps. As expected, the use of SMDS on the CAG at the destination edge eliminates the bottleneck, by moving the distribution point beyond the access link.

TABLE 1. Bandwidth measurements using 1.5Mbps encoded video stream

Client #	Avg. Bandwidth Unicast (Mbps)	Avg. Bandwidth SMDS (Mbps)
1	1.5	1.5
2	1.3	1.5
3	1.0	1.5
4	0.5	1.5

4.2 Dynamic Content Adaptation Service

A wide choice of web access devices such as wireless phones, televisions, PDAs and PCs allows almost ubiquitous web connectivity. The wide ranging display capacities of these devices result in inefficient rendering of content originally intended for display on a PC monitor (typically a 15"-17" SVGA screen). As the types of access devices in use increases, content providers are faced with the costly proposal of replicating content in multiple formats, a scenario that can quickly become a content management nightmare. Another attribute that varies among access technologies is the access bandwidth such as wireless access (from 9.6 kbps to 128 kbps), dialup modem (up to 52 kbps) and broadband access (in the order of Mbps). Large sized content, such as graphics-rich web pages, results in increased download times over slow access links. We present the Dynamic Content Adaptation Service (DCAS) that executes on the CAG edge node and dynamically customizes content for presentation, personalization, or transportation [27].

The Dynamic Content Adaptation Service is deployed on the CAG node at the edge of an internet service provider (ISP) access network or the content provider (enterprise) network. The service performs two functions: compression and content manipulation for presentation. Compression is based on the link speeds between the user and the ISP network. Content manipulation is based on the device display capacities. These pieces of information are made available to the service in the form of user profiles when the user subscribes to this service. This service presents significant commercial potential as it eliminates the need for content replication and simplifies the management of content for the content provider while enhancing the user experience.

We developed and tested the DCAS on our CAG prototype testbed shown in Figure 4. Apache web servers on PCs were used for the content provider. The clients were regular PC end systems. We deployed the DCAS on only one destination CAG edge node in our experimental enterprise network while the CAG node at the source edge simply acted as an IP router. For the initial proof of concept, the DCAS service only compresses JPEG graphics images.

4.2.1 Operation Details

The DCAS service on the destination CAG operates as follows:

1. Once the DCAS is downloaded to the CRM on the CAG and enabled, it sets up a filter on the Alteon switch via API calls, which we will call the request filter. The request filter traps request (HTTP GET) packets originating from any user to the content provider based on the Destination IP address of the web server and Destination port 80 (HTTP). The action when the request filter is triggered is set to redirect the traffic to the DCAS on the CRM via a tunnel. The IP address of the content provider is known ahead of time based on contractual agreements between the ISP and the content provider.
2. The DCAS software application then binds its receiver function to this tunnel and suspends, awaiting the filter to be triggered.
3. When a user attempts to connect to the content server, the request filter is triggered and the request packets arrive at the DCAS. The DCAS can authenticate the user and check subscription information and preferences as well. It registers the user into a local database. If the user has subscribed to the service, the DCAS sets a user-specific content filter on the Alteon switch. The content filter traps any content packets originating from the content server to that user based on Source IP address, Destination IP address and port number. The action for this content filter is also set to redirect the content packets to the DCAS on the CRM.
4. The DCAS software then forwards the client connection requests on to the web server and awaits the response.
5. When the server starts sending the web content back to the user, the content filter on the CAG is triggered and the content packets are redirected to the DCAS.
6. The DCAS software then looks for JPEG images embedded within the content payload and compresses the images if required based on the client profile (access bandwidth and access device type). The compression factor (high, medium, low) depends on a selection made by the user during subscription.
7. The DCAS software then forwards the content onto the user that originally requested the content.
8. The images downloaded to the user device are displayed on the web browser.

For the purposes of the experiment, we limited the functionality of the DCAS to JPEG transformation. The DCAS implements the JPEG standard which uses the Discrete Cosine Transform as the heart of its compression algorithm. The DCT function involves performing matrix multiplications and is well understood to be a compute intensive task for CPUs. In order to enhance performance of the DCAS, we implemented the DCT function in hardware on the FPGA-assisted CE module.

4.2.2 Performance Measurements

We conducted experiments to evaluate the performance of DCAS on CAG by varying the size of the JPEG images, compression factor and processing granularity. The content used for testing involved a simple web page with one or two JPEG images. The DCAS service was deployed on the CAG edge node at the destination i.e. close to the user. To eliminate any variations resulting from the Internet, we assumed that the content was available at a cache at close proximity to the CAG node. We used a 100Mbps connection between the cache node and the CAG node. Table 2 shows the download times for a web page on various low speed links that simulate wireless, dialup and ISDN rate data access. The 33Kbyte sized web page used for the test included 2 JPEG images of 16Kbytes each. Using a high compression factor we achieved a 4:1 compression ratio. The impact on download times are as expected. The presentation aspect of the DCAS service is a qualitative measurement that depends on the expectations and preferences of the user and was therefore left out of the experiment.

An important metric for a Content-aware Active Gateway is the number of sessions that can be processed per second. Unlike routers, where wire-speed performance of the well-defined routing/forwarding function is a key requirement, the performance of a CAG depends on the type of services supported by the CAG at any given time. In many cases, wire speed performance for these services is impossible to achieve using current technologies. For the DCAS service, a session refers to the transfer of an average sized web page.

Due to the wide variances in web content, it is important to define a common metric that can be used to compare performance across applications as well as platforms.

TABLE 2. Effect of DCAS on 33Kbyte web page download times

Link Speeds (Kbps)	Download Time (sec)	Download Time with DCAS (sec)
4.8	60	15
9.6	30	8
28.8	10	3
52	5	2
128	2	1

5.0 Discussion

This section presents some of the lessons we learned through our research and development efforts described in this paper. The SMDS and DCAS content networking applications have a common underlying flow. This high-level flow, shown in Figure 6, can be used as a template for a wide range of content networking applications. The most important block in the flow, the Content Processing block, forms the heart of the application. In the SMDS, this block performs streaming content replication and in the DCAS, it performs the image compression. Blocks performing the Authentication, Authorization and Accounting are necessary in a real world commercial service.

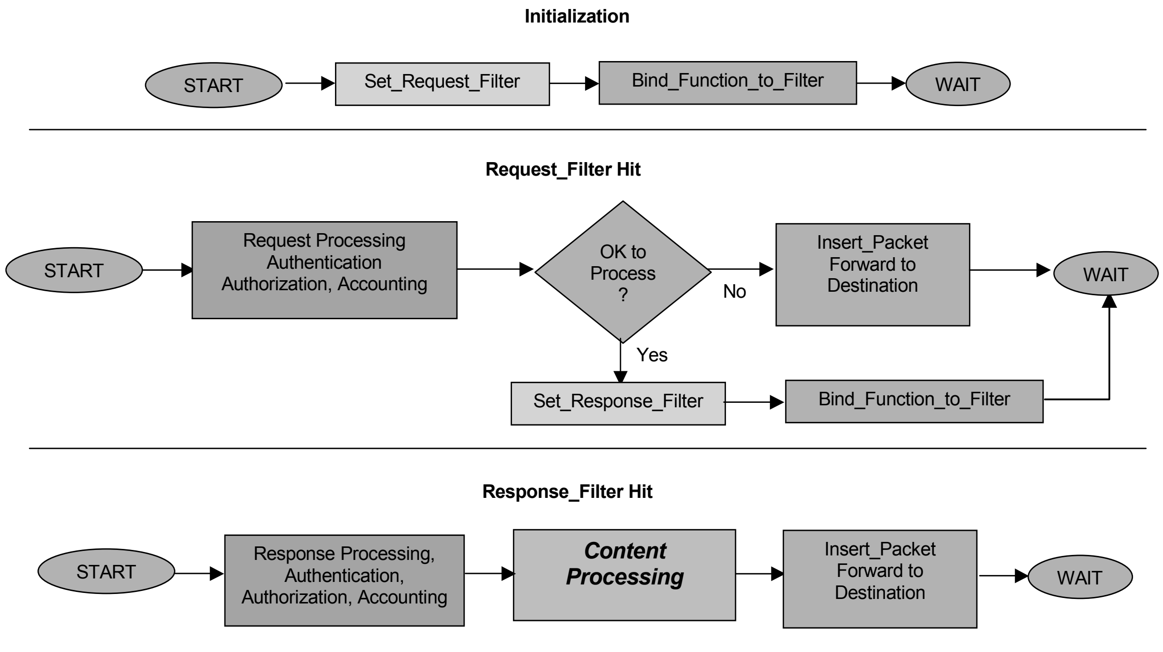

Figure 6: Content Networking Application High Level flow

We found that hardware processing is of utmost importance for providing real world performance to content networking applications on a CAG node. This became apparent during our SMDS experiment where we found that the SMDS on the CAG was unable to support multiple Real video clients when the encoding rate of the video stream was increased to 5 Mbps. Investigation using the profiling tool, gprof, revealed that the duplication and checksumming functions within SMDS took about 40% of the CPU time and the Pentium based CE module could not handle the increased processing requirements of multiple clients viewing high bit-rate video content. We are working on using custom hardware to perform these functions on the FPGA-based CE module and will present these results in future publications.

It is important to select the right level of granularity for content processing applications as it can have an impact on latency, memory requirements and performance. For instance, in the DCAS service which compresses images on-the-fly, the granularity refers to the number of image blocks to be processed before switching to image blocks of another image or user. Given that the service can encounter simultaneous multiple images from a single page or from multiple users, the granularity size decides the amount of memory for the queues required to store the image blocks awaiting processing. In content networking applications this requirement can increase drastically with the number of users. Hardware assisted content processing can alleviate memory requirements by increasing the rate at which the image blocks are processed.

6.0 Related Works

A significant amount of research has been published in the area of intelligent network services through programmable networking, ranging from networking paradigms and re-programmable hardware to application environments. [1,3,6,15,21,22] The Active Networks approach [2,4,5,7,9-11] is a major effort to incorporate programmability into the network infrastructure. To date, these developments have been mainly realized in software-based hosts (e.g., Linux-based systems) that offer the required programmability but lack the performance required in real networks. The Washington University ANN (Active Network Node) [9,19] implementation introduces an FPGA-based CPU module that accommodates the active code into a high performance routing switch. Programmable hardware for routers is explored in [3,4,6,13]. Other works such as Darwin [8], Phoenix [17] and Openet [1,6,9,13,15,19] have investigated mechanisms for delivering programmability to end-users. The Openet effort [15] describes a use of a real-world routing switch to achieve programmability at a high level of forwarding performance. However, the implementation does not provide for scalability of computation performance as it uses a single control plane CPU to perform the computations. Unlike existing research in Active Networks which are primarily focused on active routers, we take a practical approach at incorporating active networks research concepts into network gateways. Our efforts towards the CAG architecture, described in this paper, are centered around the requirements of commerical networking devices and focus on effective transfer of active networks concepts to real world networking devices.

7.0 Conclusion

We have presented a new application for active networking concepts in the form of a Content-aware Active Gateway. The CAG at the network edge is an important concept in the field of active networks as it represents a significant step towards realization of active networks research concepts in real-world networks. We justify the introduction of the CAG at network edges by demonstrating two practical services with near-term potential, SMDS and DCAS. The increase in the complexity of the Internet architecture at the network edges, due to the number of commercial content networking applications that have been introduced over the last few years, also justifies the need for the CAG. The software architecture that we have developed uses key Active Networks concepts to enable service creation and dynamic service deployment on the CAG. We have implemented a CAG protoype using real-world high-performance networking devices augmented with software that we developed. Our efforts indicate that our CAG architecture can support the requirements of a wide range of content networking applications. Performance requirements of content networking applications are satisfied through the use of FPGA based hardware acceleration within the CAG processing modules. Further research and development efforts are planned to investigate performance requirements for CAG in real networks supporting multiple services and processing thousands of flows simultaneously.

8.0 References

1. K. Yamada et.al., "A Novel Approach to Realizing Flexible Transport Systems using Reconfigurable Hardware", Proceedings of the IEEE Symposium on FPGAs for Custom Computing Machines, April 1995, pp.67-71.
2. D. Tennenhouse, D. Wetherall, "Toward an active network architecture," Multimedia Computing and Networking, Jan 1996.
3. Y. Yemini, S. Da Silva. "Towards Programmable Networks", IFIP/IEEE Intl. Workshop on Distributed Systems: Operations and Management, L'Aquila, Italy, October 1996.
4. D. L. Tennenhouse, et al, "A Survey of Active Network Research", IEEE Communications Magazine, Vol. 35, No. 1, January 1997
5. S. Bhattacharjee et al., "An Architecture for Active Networking," Proc. INFOCOM'97, Apr. 1997.
6. I. Hadzic, J. Smith, "On-the-fly Programmable Hardware for Networks", Proceedings of GLOBECOM 1998.
7. D. J. Wetherall, J. Guttag, D. L. Tennenhouse, "ANTS: A Toolkit for Building and Dynamically Deploying Network Protocols", IEEE OPENARCH'98, San Francisco, CA, April 1998.
8. P. Chandra et al, "Darwin: Resource Management for Value-Added Customizable Network Service", Proc. 6th IEEE ICNP, Austin, Oct. 1998
9. D. Decasper, et al, "A Scalable High Performance Active Networks Node", IEEE Network Magazine. Vol 37, Jan/Feb 1999, pp 8-19.
10. B. Schwartz, A. Jackson, T. Strayer, W. Zhou, R. Rockwell, C. Partridge, "Smart Packets for Active Networks", IEEE OpenArch 99, New York, March 1999
11. J. M. Smith, K. L. Calvert, S. L. Murphy, H. K. Orman, L. L. Peterson, "Activating networks: A progress report," IEEE Computer, Vol. 32, pp. 32-41, Apr. 1999.
12. T. Lavian, R. Jaeger, J. Hollingsworth, "Open Programmable Architecture for Java-enable Network Devices", Stanford Hot Interconnects, August 1999.
13. G. Hjalmtysson, S. Bhattacharjee. "Control-on-Demand: An Efficient Approach to Router Programmability," IEEE Journal on Selected Areas in Communications, Vol. 17, No. 9, Sept. 1999, pp. 1549-1562.
14. I. Stoica, T.S.E.Ng, H.Zhang, "REUNITE: A recursive unicast approach to multicast", IEEE INFOCOM'2000 Mar.2000.
15. T. Lavian, P. Wang, "Active Networking On A Programmable Networking Platform", IEEE OpenArch'01, Anchorage, Alaska, April 2001
16. T. Lavian, P. Wang, F. Travostino, S. Subramanian, D. Hoang, V. Sethaput, D. Culler, "Enabling Active Flow Manipulation in Silicon-based Network Forwarding Engine." IEEE Journal of Communications and Networks, March 2001, pp.78-87.
17. D. Putzolu, S. Bakshi, S. Yadav, R. Yavatkar, "The Phoenix Framework: A Practical Architecture for Programmable Networks", IEEE Communications Magazine, Vol 38, No 1, March 2001.
18. Y.H. Chu, S.G. Rao, S. Seshan, H. Zhang, "Enabling Conference Applications on the Internet using an Overlay Multicast Architecture", SIGCOMM'01, San Diego, 2001.
19. T. Wolf, J. S. Turner, "Design Issues for High-Performance Active Routers", IEEE Journal on Selected Areas in Communications, Vol. 19, No. 3, March 2001, pp. 404-409.
20. L. Henrique, M.K. Costa, S. Fdida, Otto Carlos M.B. Duarte, "Hop By Hop Multicast Routing Protocol", SIGCOMM'01, August 27-31, 2001, San Diego, California, USA.
21. T. Lavian, P. Wang, F. Travostino, S. Subramanian, D. Hoang, V. Sethaput, "Intelligent network services through active flow manipulation", Intelligent Network Workshop, 2001 IEEE , 2001, pp. 73 -82.
22. D.E. Taylor, J. S. Turner, J. W. Lockwood, "Dynamic Hardware Plugins (DHP): Exploiting Reconfigurable Hardware for High-Performance Programmable Routers", IEEE Openarch 2001, pp. 25-34.
23. B. D. Davidson, "A Web caching primer", IEEE Internet Computing,, Vol. 5, Issue 4, July-Aug. 2001, pp. 38-45.
24. J. S. Chase, "Server switching: yesterday and tomorrow", WIAPP 2001, Proceedings of the second IEEE workshop on Internet Applications 2001, pp. 114-123.
25. J. Lu, "Reactive and proactive approaches to media streaming: from scalable coding to content delivery networks", Proceedings of International Conference on Information Technology: Coding and Computing, 2001. pp. 5-9.

26. T. Lavian, P. Wang, R. Durairaj, D. Hoang, "Application Layer Multi-Unicast from Edge Device," Submitted for publication.
27. S. Subramanian, G. Boissonnard, C. Gloster, J. Rasimas, "High-Performance Configurable Content Networking", Submitted for publication.
28. Nortel Networks Inc., www.nortelnetworks.com
29. Volera Inc., www.volera.com
30. Akamai Inc., www.akamai.com