Technical Report 2002-03-02
Internetworking and Media Communications Research Laboratories
Department of Computer Science, Kent State University
http://medianet.kent.edu/technicalreports.html

UBIQUITOUS INTERNET APPLICATION SERVICES ON SHARABLE INFRASTUCTURE

JAVED I. KHAN and YIHUA HE

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
March 15, 2002

Abstract

In this paper we present an overlay infrastructure for provisioning content services over the internet. This provides a uniform framework to develop ubiquitous application services and deploy it on shared infrastructure on need basis. The custom application service modules can flow to the service optimum network vicinity and if necessary into multiple network points for composing co-operative services. The resulting framework will be able to support a wide array of emerging application service scenarios, ranging from simple caching to content customization, content adaptation, localization, ubiquitous computing to co-operative filtering, content-mining, to high performance composite multimedia transcoding.

Key words: application service, active processing, content services networking.

1. Introduction

Application services networking will take a center stage in the internet based applications research in coming years. We are already seeing the emergence of *content delivery networks* (CDNs). A CDN can be viewed as an overlay network of customer content, distributed geographically to enable rapid, reliable retrieval from any end-user locations. Caching proxies, which are normally deployed at the edge of CDNs, play important rolls in duplicating and delivering the content to the end-users. Originally, these caching proxies are not supposed to modify the content. However, with the deployment of the first systems it seems that the CDNs infrastructure can be extended to provide customized content delivery. A look at any modern portal will show the demand. Today's web content is no longer mono-modal. It is not only a simple HTML with few embedded images. A webpage typically consist of documents of multiple modalities: audio, video, text. Also another growing trend is that contents for served documents are actually coming from multiple parties. As the Internet continues to evolve with increasing diversity and heterogeneity, we are seeing a growing demand for additional services such as content adaptation, personalization, watermarking and location-aware data insertion [1]. However, such content adaptation services are only a small picture of more general content services which can be provided in near future.

It seems a radical generalization of the concept of cache or proxy is overdue. In future it (or a modified version of it) will assume much more active role. Rather than being only a temporary cache of HTTP responses will be increasingly capable of arbitrary processing of the incoming information. It has the potential to be used in much more innovative ways. These can potentially act as a hub for various actions ranging from rich domain knowledge based information steering, filtering, multiplexing, adaptation that will be required by ubiquitous services.

In this research we pursue the novel concept of overlay networking for ubiquitous *Internet Application Services*. One can think IAS is an overlay network of sharable application proxies, who have computation resources to perform value-added services for the content-provider and the end-user.

The potential benefit of such internet application services are so pressing that most modern sites are already simulating them in various ways. However, due to the lack of any sharable infrastructure, currently most adaptations are performed in content provider's own site typically with array of backend special purpose application processing servers. Typically the content provider has to develop and own both the application service and the servers.

This model is adhoc and has several quite paralyzing limitations. A majority of the services, such as end-user's device-based adaptation, localization, distribution, content transcoding for wireless environment, all naturally require actions inside network where the topological and geographical placement of the actions can be a very important consideration. Also, the placement of services upstream from the embedded caches invalidates caching in most cases. Also, the content-server site based processing inherently limits the load distribution and fault-tolerance. There is also software engineering drawbacks. Many application services will require specialized know-how. Considering the complexity and the cost of long term maintenance to keep services competitive and up-to-date it for most complex services it will be best if done by specialized third parties.

In this paper we present a vision, and a proposal towards a content services networking that will allow separation between the application service provider, the content provider, and the platform provider, and define a framework for the systematic and secured interplay. The framework can supports deployment of wide range of services with various specification and initiation dependencies. In the next section we briefly present some of the very recent and interesting developments in this fast unfolding area.

2. Content Services Networking

2.1. Caching to CDN

The research in distributed caching over past five years has evolved into *content distribution networks* (CDN). Commercially, we have seen the emergence of global content caching systems such as Akamai [8]. More recently, a number of teams are looking technology for content adaptation at an origin server or in these proxy caches. Example works include Spyglass[5], ProxiNet[6], Intel QuickWeb[7], IBM Transcoding proxy[10], UC Berkeley TranSend[11], Digestor[12], Mobiware[13], and Smart Client[14].

Active Proxy: For provisioning of value added service in the caching proxies the IETF Working Group has recently proposed the Open Pluggable Edge Services (OPES) [19] and the Internet Content Adapation Protocol (iCAP) [20] defining the basic functions of future caching proxy. iCAP defined how various caching objects can be transported from one cache to another. OPES provides some interesting capabilities to caching proxies. An OPES proxy can be equipped with message parsers, rule modules, and proxylets library. When messages flow through an OPES proxy, not only it is cached but they can also be automatically parsed and processed with these rules. The OPES proxylets can execute the processing in the caching proxies, or optionally can call for remote callout services via protocols such as iCAP [2]. The OPES is overly caching proxy centric, where it not only converts the documents, but also takes part in coordinating network elements. And almost all active processing logic in OPES comes from the interplay of the message content and the rules in the proxylets. This approach does not provide enough flexibility in accommodating various service arrangements that may arise in the real service deployments, which often restricts where, when and how the service can be performed, redirected, and who may provide the service specifications.

2.2. CDN to CSN:

Ma et. al. [3] suggested an enhanced model of *content services networking* (CSN) takes a more application server (or proxy) oriented view. Ma's CSN separates caching proxies from application servers. The application servers can directly communication with the content servers and user-agents to position itself in the message pathway. Ma shows indeed this approach can handle more flexibly more complex service scenarios. These include post or pre distribution services either on behalf of the user agent or on behalf of the content-provider. Also, it allows for more versatile services to be placed into the system as the processing is performed entirely in the application proxy—a separate entity than the caching proxy. Also, a notable advantage of Ma's proposal is that the 'application service' here does not mean only XML-like markup languages processing such as dynamic assembly and delivery of web content. Ma shows an example application service which can be an interactive Video Delivery processing [2], which indeed can be very cumbersome if done via OPES like declarative rules. Indeed, majority of multimedia content protocols are highly compressed and domain specific data structures, and can only be superficially 'processed' with rules.

However, Ma's CSN model requires the traffic to be diverted to the application service provider's (ASP) custom servers. The infrasutrcture application proxy will be located in application service provider's owndership domain. This is a serious limitation. Within this model essentially all *application service provides* (ASP) would have to have their own worldwide infrastructure. This is what happened with Akamai[1] [8].

In this backdrop we propose an application services network, which can retain the advantages of both the approaches. We further refine the CSN concept and present a strong application service oriented approach rather than application server oriented approach of Ma et. al. We further separate the application server from the application service. We envision OPES proxy like sharable platforms to be available from the infrastructure providers. The important distinction is that in this model the CSN infrastructure provider does not have to get involved with the details of the CSN service. The CSN infrastructure provider can concentrate on the provisioning of CSN servers at topologically and geographically strategic locations in the network. On the other hand a CSN service provider can focus on developing the service, without worrying about the infrastructure. We propose the *active service distribution and location* (ASDL) model that will allow these two parties to work together and provide application service between the end-user and the content-provider. This will not require traffic to be diverted into application service provider's site rather the application service provisioning modules will flow into the locations in their natural path which are best with respect to strategic consideration—including even content server's location.

3. Active Networks

An important piece in such location independent internet application service provisioning is the sharable code servers inside network. In recent years some very interesting advances have been made in programmable networks. Among them active networks [25, 26] initiative proposes the generalization of the traditional router concept— where transiting

[1] Akamai's currently runs about over 14,000 servers globally.

packets can be modified almost in any way with custom embedded program modules in the network elements. At least eight architectures have been proposed explored with great promise. Examples include ANTS [25], SmartPacket [21], PLAN [23], NetScript [23], LIBRA [17], DARWIN [16], and our Virtual Switch Machine [24]. A number of issues have been explored and several potential OS architectures have been proposed (including one of ours [24]). In this research we investigate a quasi-active router like devices can be eventually used for ubiquitous application service provisioning.

To summarize--- this approach has the following novel aspects:

- The application service provider does not need to have their own worldwide application server infrastructure.
- There can be geographically distributed hierarchical (ISP like) CSN server providers. Provides will be able to choose value added service from multiple application service providers for their customers.
- Provisioning of application service will not require traffic to be artificially diverted to custom ASP servers. Rather, the modules will flow to the appropriate network points.
- The service modules can also flow to multiple network points for co-operative processing, including at content server site. This will enormously expand the range of application services which can be handled by this scheme.
- The specialized ASP processing servers allows the supported services to work into deeper network layers (such as TCP/IP). This will provide another degree of freedom on the range of applications services that can be deployed. This will also enables more efficient implementation of upper layer services.

4. Active Service Distribution and Location Model

2.1 *Architecture and components:*

The Active Service Distribution and Location (ASDL) Model we propose identifies the following three major components as shown in Figure 1. They are:

1. Active Routers (AR): AR hosts the software of value-added services, which are normally known as one of the tasks of an ISP (Internet Service Provider). These active routers are sparsely distributed special stream computing platforms typically deployed near the edge of the Internet backbone. These can be owned by certain ISPs (or overlay ISPs). In

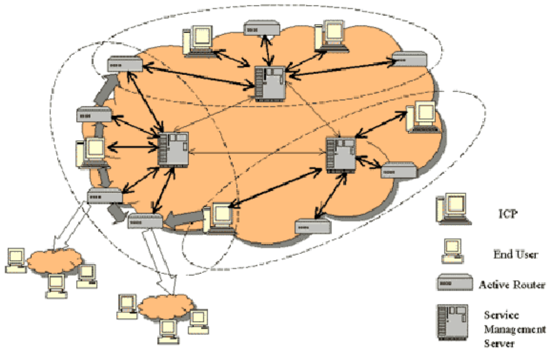

Fig 1: ASDL Architecture and Components

addition to the normal routers' functions, these ARs can provide computational resources to process data streams passing-by, either on behalf of content providers (CPs) or on behalf of the end users. If the service is on behalf of the ISP itself, we consider it is on behalf of the end users. Unlike the CSN's application proxy servers [1], ARs finish most value-added service at application as well as in IP levels. These routers have special TCP/IP layers, which enable them to fast intercept streams. The processing speed can be much higher than in application level, because (1) much less decapsulation, encapsulation work will be needed; (2) and simpler instructions in IP level will let us take advantage of RISC technology; (3) and some of data streams may indexed or marked by the corresponding ICP serverlet, which means they can be facilitated with the ability of random access in the data stream. See [27] for an architecture of such a system. With the pattern dependent dynamic pointing by content providers' serverlet, the applications built on top of the active routers can add, delete or modify the content of the stream randomly, depending on the specified data pattern. The active routers have a variety of operational modes depending on how the service is performed. We are going to investigate them in section 2.2.

2. Service Management Servers (SMS): SMS serve as the principle service provider. It acts as the mediation center among the end-users, active routers and the content providers. The SMS servers own the modules called *switchlets* which are dynamically deployable to the ARs. These programs form the actual service. SMS servers are responsible for the following tasks: (1) they maintain static and dynamic information about the service execution environment and the locations of the applications; (2) they receive the service registration or cancellation requests from end-users, active routers or content providers, and choose to accept them or not according to authentications; (3) After initial service registration, SMS servers can choose to accept or reject a service deployment request by measuring the availability of the service and its registration status; (4) SMSs aggregate the information about usage, availability and location of each deployed service, and then provide the information back to the deployment requester. (5) SMS also provides management, dynamic status visualization and monitoring, accounting and billing functionalities to value-added-in service participating parties who use ASDL as an information exchange path. (6) Each SMS is responsible for collecting information about its domain and periodically exchanges the information, such as registration and deployment status, with other cooperating SMSs. These exchanges can be triggered automatically if there is a change in the system.

3. Content Providers (CP): are where the original data streams come from. These CP servers can be typical web servers. However, the protocol allows *servelets* to be deployed at sender's location, if required by any service. For example a serverlet may premark the outgoing data streams, when a particular service is active on the stream. Not every streams coming from the CP will have a marker. In most cases, streams with and without markers are mixed, even from the same CP. The marker in a data stream has two purposes: (1) differentiate the data stream that needs service from that need not; (2) make random access available to Active Routers (ARs), and therefore dramatically reduces the computation burden of ARs. We will show an active hyperlinking example in section 4 and the saving and the cost will be shown and analyzed in our example in section 4.2. Servelets may come from CP itself or from value-added service providers, but they must be registered in the SMS of its own domain, and deployed with proper authentications. The proposed protocol will perform these seamlessly. Active routers may receive data stream with marker, but end-users will not. This is because the markers, if any, have been taken off after processed by ARs, and only normal data streams will be sent to end-users.

4. End-users: are the sinks/terminals of data streams. They may be the normal desktop/laptop computers, or maybe handheld or wireless devices, or wearable computers. These terminals may have some kind of resource limitation, and therefore they need the resource or service provided by the ISP/AR. For example, a laptop computer hooked up to Internet by a telephone line modem cannot sustain high quality video stream. In order to make the video viewable, the quality of the video must be gracefully lowered at a certain AR. In our model we envision normal end-users. However, we also provision for specialized browsers which might be able to supply additional information about personalization services.

5. ASDL Protocol

For every internet content service, there is a service initiator who initiates the service by sending the service request to SMS and other service parties. After information is exchanged several times among these parties, the service is deployed. However, some services may require further parameters. These parameters can again be of two types static and dynamic. The static adaptation parameters are those which can be received before the service begins. The dynamic adaptation parameters are those which are required with every request. We call this kind of parameters as *service dependence specifications*, and the party who send out the specifications as the specifier. For example, a content provider may initiate an MPEG stream and request a service, which will transcode the stream according to the bandwidth of the end-user. At this time, an end-user may specify a certain bandwidth and send this information to the service provider. (If the bandwidth is not specified, the service provider may choose a default bandwidth.) In this case, the content provider is the initiator,

Specifier	Destination of specification		
	EU	**CP**	**AR(SP)**
EU	No	Yes, by HTTP extensions or web forms	Yes, by HTTP extensions or web forms
CP	Yes, by XML or HTTP meta extensions	No	Yes, by serverlet
AR (SP)	Yes, by HTTP meta extensions	Yes, by active application program	No

Chart 1: Specification method between different parties in the service

and the end-user is the specifier.

ADSL also allows dynamic custom index based random stream access. A serverlet running on the content provider's site is a program that can be designed to help the service from the content source, such as source file indexing. An active application is a program which provides the service directly to the end-user, and it is designed to run on an active router, which is normally controlled by some internet service provider (ISP). Before we introduce the ASDL protocols, we'll first consider three issues: (1) who is going to supply the serverlet running on the side of content provider and the active applications running on the side of service provider? (2) Who is the service initiator? (3) Who is going to provide the dependence specification of the service?

In our model all servelets and switchlets comes from an application service provider. A single service may require switchlets and servelest to be deployed into multiple points.

All the three parties (EU, CP and AR) can be initiators and dependent specifiers of the ASDL services. However, when the initiator and the specifier are the same party, there is no need for extra transmission of the specification, since the dependent specification can be done at the initiation stage. For example, if an end-user is requesting a bandwidth adaptation service, he or she can include the bandwidth information inside the initial request. However, transmission for dependent specifications between different parties is necessary. There are several ways to transmit the specifications: (1) by tightly coupled serverlet and active application programs (2) by XMLs or XML-like languages (3) by meta tags. The specifications between a CP and a SP can be expressed by method (1), because they share a couple of servlet and active application programs, both of which derived from SMS. Information can be exchanged freely between the coupled programs. The specification from content providers to the end-users can be expressed by the XMLs and HTTP meta extensions, while the specification from AR(SP) to the end-users can only be expressed by the HTTP meta extension. The CP and AR can make up web forms for the end-users convenience to provide the specification information. Chart-1 summarizes the discussion.

Example of Active Services	Mode		
	Single Service	Group Service	
	EUI	SPI	CPI
Insertion of Ad Banners		*	
Multimedia adaptation for limited client bandwidth	*	*	*
Multi-language adaptation for different user preference	*	*	*
Active hyperlinking	*	*	
Active re-direction	*	*	
Virus Scanning	*		
Stream data adaptation and optimization	*	*	
Watermarking			*
Insertion of regional data	*	*	
Language translation	*		

Chart 2. A list of example services and the modes they belong to

From the service requesters' view, we may classify the services into two categories: (1) the single service request and (2) the group service request. A single service is requested by a single user and it will work solely for one user to meet its specific request. For example, a handheld device holder may request the active router to translate all English web pages into German. This cannot be done at the handheld device, since it lacks memory, storage or processing speed to finish that task. In the case, the end-user may "buy" computation resource from the "net". The other type of service is group service, which is initiated either by the service provider or the content provider. For example, a service provider may have some agreement with the third party and advertise for them. The service provider then can analysis the web html files and put the ads at appropriate places. The group service can also be initiated by content providers. For example, a video source server may put special marks in the video stream and help the active routers to downscale the video gracefully and meet the bandwidth requirement for all different users. The service examples and the modes they belong to are listed below in chart 2. Group services may require additional information exchange between different parties, as we discussed in chart 1. The example of additional information exchange may include the user's bandwidth limitation, the configuration of user's browser and etc.

For the specifications the specialized devices and content servers generally maintain a *personalization cookie box* which contains a set of tablets containing the user, user-agent, and user-environment specific constraint information. ADSL provides communication steps for exchange of these information.

Below we now explain the ADSL model by describing how it handles the specific service scenario for three different application service models.

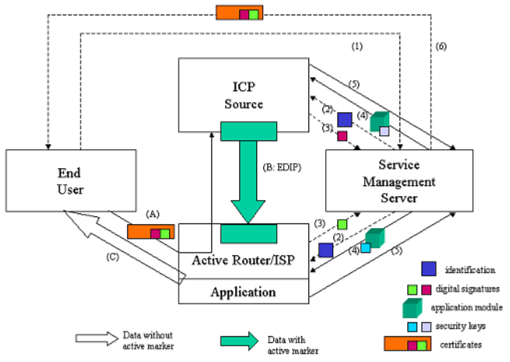

Fig 2: End-user initiate single service

5.1. EUI model

In this scenario, the end-user initiates the service. Fig-2 illustrates the communication steps.

Setup Stage:

(1) The End User (EU) sends service request to Service Management Server(SMS). (2) SMS sends query to the participating ICP Source (ICPS) and Active Router(AR) to collect necessary configuration data. The query is with the identification of the SMS. (3) The ICPS and the AR response with digital signature for authentication and other necessary configuration information to SMS (4) SMS then delivers the application modules

to ICPS and AR, with corresponding security keys, which are required when installing the modules
(5) The ICPS and the AR send back the acknowledgements.
(6) MSM sends the response back to EU with the certificates that EU may need when sending requests to AR and ICPS.

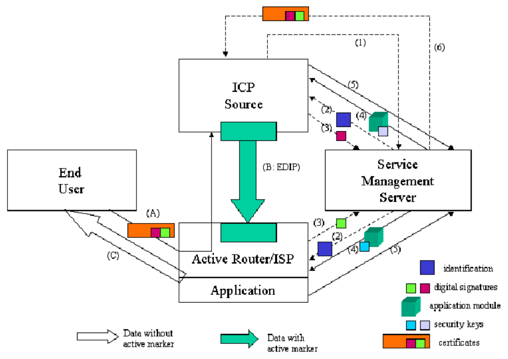

Fig 3: Content provider initiate group service

Data Transfer Stage:

(A) EU sends the data request with certificates provided by MSM.
(B) ICPS sends out data packages with EDIP headers.
(C) AR processes the packages with EDIP headers, performs value-added in service, and sends result to EU with normal IP packages

5.2. CPI Model

In this scenario, the content-provider initiates the service. Fig-3 illustrates the communication steps.

Setup Stage:

(1) The ICP Source (ICPS) sends service request to Service Management Server(SMS).
(2) SMS sends query to the participating ICP Source (ICPS) and Active Router(AR) to collect necessary configuration data. The query is with the identification of the SMS.
(3) The ICPS and the AR response with digital signature for authentication and other necessary configuration information to SMS
(4) SMS then delivers the application modules to ICPS and AR, with corresponding security keys, which are required when installing the modules
(5) The ICPS and the AR send back the acknowledgements.
(6) MSM sends the response back to ICPS with the certificates that ICPS may need when sending requests to AR.

Data Transfer Stage:

(A) EU sends the data request.
(B) ICPS sends out data packages with EDIP headers.
(C) AR processes the packages with EDIP headers, performs value-added in service, and sends result to EU with normal IP packages

5.3. SPI Model

In this scenario, the service provider itself initiates the service, and requests contracts from the content provider and active routers. Fig-4 illustrates the communication steps.

Setup Stage:

(1) The Service Provider (SP) sends service request to Service Management Server (SMS).
(2) SMS sends query to the participating ICP Source(ICPS) and Active Router(AR) to collect necessary configuration data. The query

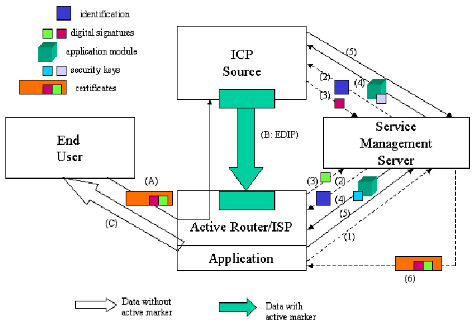

Fig 4: Service provider initiate group service

is with the identification of the SMS.
(3) The ICPS and the AR response with digital signature for authentication and other necessary configuration information to SMS
(4) SMS then delivers the application modules

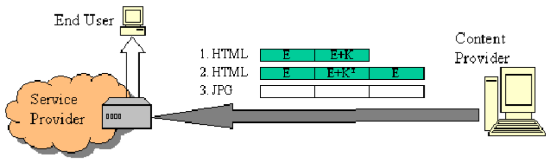

Fig 5: The example active hyperlinking service in our ASDL model.

to ICPS and AR, with corresponding security keys, which are required when installing the modules
(5) The ICPS and the AR send back the acknowledgements.
(6) SMS sends the response back to ICPS with the certificates that ICPS may need when sending requests to AR.

Data Transfer Stage:

(A) EU sends the data request.
(B) ICPS sends out data packages with EDIP headers.
(C) AR processes the packages with EDIP

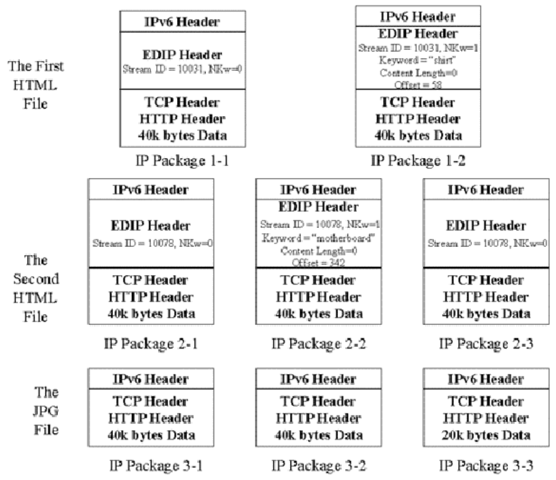

Fig 6: The three files have been requested by the end-user, after marked by the serverlet

headers, performs value-added in service, and sends result to EU with normal IP packages.

6. Example and analysis

Finally, we explain the operation of the system. In the process of we use a novel service called *active hyperlinking filter* [27] as our example Active Hyperlinking Filter is a service that the service provider analyses the HTML web page streams passing-by, and adds appropriate hyper links at appropriate places, upon the request from the end users or advertisement companies. It belongs to EUI or SPI. Also, it demonstrates how multi-point processing can help (in this case one active router and the service specific processing at the content server). This example will also show how this application level processing can be accelerated by low level processing at the active router. The traditional solution for hyperlinking is that the application server will decode every stream passing by and search the keyword inside. If there are some keywords found, the stream will be modified to add the hyper link into the web page. And then the content will be encoded again and send to the end-user. There are two major disadvantages: (1) Service providers have to decode the stream in order to analysis the content. (2) Service providers have to search inside the stream to find out which stream should be modified and they have to search to know exactly where the modification should happen.

By using the ASDL protocol, we can avoid those two disadvantages. The service provider contacts the SMS server and deploys the serverlet on the content server. The serverlet works as an index maker and tells which stream contains the keywords and the offsets about where they are. This information is stored in a special designed header in IP extensions. [EDIP] When active routers got the information, they don't need to decode or search, and they can process the modification immediately.

Figure 6 shows the example of hyperlinking. The service provider initiates the deployment and it contacts SMS to distribute the serverlet to the content provider before transmitting the data. In this example, the end-user is requesting 3 files (see Figure 6) --- two are html files and the other one is jpg file. The two HTML are bearing the EDIP header but only the second HTML file has the keyword in the service provider's service list. The active router of the service provider will scan the IP header extensions (EDIP header in our example) and invoke proper API to modify the content of the second HTML file. As we can see from Chart 3, in a traditional full search

Packages	Actions and Destinations (ASDL)				
	Router Level	**Intermediate**	**App Level**	**Intermediate**	**Router Level**
1-1 [E]	To buffer	▨	▨	▨	forward
1-2 [E+K]	To detector To buffer	▨	▨	▨	forward
2-1 [E]	To buffer	decapsulate		encapsulate	send
2-2 [E+K^T]	To detector To buffer	decapsulate	Modify add a link	encapsulate	send
2-3 [E]	To buffer	decapsulate		encapsulate	send
3-1	To forwarder	▨	▨	▨	forward
3-2	To forwarder	▨	▨	▨	forward
3-3	To forwarder	▨	▨	▨	forward

Chart 4: Cost and savings when using ASDL service mode

filtering service mode, all 8 IP packages have to be decoded and sent up to application level for process. However, in our ASDL powered service mode, only the IP packages containing the expected keyword will be decoded and processed later. All the other IP packages are process as normal router. This is because the original content has been indexed by the serverlet deployed on the servers of the content provider, and the indexing information is added at a special IP extension, which can be easily recognized by the active application program deployed at the service provider. With the help from content provider's serverlet, the active application does not have to decapsulate and search all IP packages in order to know where is the content we are trying to modify. Rather, the active application program can pin point where the modification is needed, simply by screening the special IP extension headers.

By this way, a lot of decapsulating, encapsulating and searching time has been saved. The shaded area in Chart 3 and Chart 4 are the actual savings by using ASDL service model, when comparing to the traditional service model.

7. Conclusions

In this paper we have outlined the ADSL proposal for facilitating content services deployment over the internet. This extends the OPES by allowing much more active management of application service. Also, this extends CSN by not forcing the application service management and execution to be co-located. Also, this proposal will allow building complex services which may need multiple but coordinated service points. A key advantage of the scheme is that it focuses on the service itself. This will enormously expand the range of application services which can be handled by this scheme. A key advantage of this approach is that services can be managed and monitored from a single view point. This will increasingly become important as the number of modules and servers grows. We are currently implementing the SMS server which will allow seamless management and monitoring of any deployed service. It also includes dynamic reporting protocol, where individual service components will send back dynamic information to the SMS for its operational status.

In this research we have just outlined the potential extensions to the OPES like protocols. We are also currently defining schemes when a single service over ASDL can be deployed into multi-domain CSN servers with separate ownership. A point to note is that many of the issues of CSN will be associated with the definition of component and content ownership in a complex multiparty information distribution and en-route processing framework within legal and social constraints and implications. This indeed will dictate the scenarios. The objective of our CSN research is to support as many of them as possible.

The work is currently being funded by the DARPA Research Grant F30602-99-1-0515 under its Active Network initiative.

8. References:

1. Wei-Ying Ma, Bo Shen and Jack Brassil, "Content Services Network: The Architecture and Protocols", International workshop on web caching and content distribution, June 2001.
2. S. J. Lee, W. Y. Ma, and B. Shen, "Interactive video caching and delivery using video abstraction and summarization," Proc. International Workshop on Web Caching and Contenet Distribution (WCW'01), Jun 2001
3. M. Hofmann and A. Beck, "Example Services for Network Edge Proxies", Internet-Draft draft-hofmann-esfnep-00.txt, Sept 2000
4. J. I. Khan and S. S. Yang, "Adaptive Netcentric Application Framework on Active Network",
5. Spyglass-Prism. http://www.spyglass.com.
6. Intel QuickWeb. http://www.intel.com/quickweb.
7. ProxiNet. http://www.proxinet.com.
8. Akami. http://www.akami.com
9. ESI. http://www.esi.org
10. J. Smith, R. Mohan, and C. Li, "Scalable multimedia delivery for pervasive computing," ACM Multimedia, 1999.
11. Fox, S. D. Gribble, Y. Chawathe, and E. A. Brewer, "Adapting to network and client variation using active proxies: lessons and perspectives," IEEE Personal Communication, Vol. 5, No. 4, pp. 10-19, August 1998.
12. T. Bickmore and B. Schilit, "Digestor: Device Independent Access to the World Wide Web", proceedings of the Sixth International World Wide Web Conference, Santa Clara, California, 1999.
13. O. Angin, A.T. Campbell, M. E. Kounavis, and R. R.-F. Liao, "The Mobiware Toolkit: Programmable support for adaptive mobile networking," IEEE Personal Communications, Vol. 5, No. 4, August 1998, pp. 32-43.
14. C. Yoshikawa et al, "Using Smart Clients to Build Scalable Services," Proc. Winter 1997 USENIX Tech. Conf., January 1997.
15. DARPA ACTIVENET http://www.darpa.mil/ito/research/anets/
16. CMU DARWIN http://www-2.cs.cmu.edu/~darwin/
17. CMU Libra http://www-2.cs.cmu.edu/~libra/
18. xS. Blake, D. Blake and etc, "An Architecture for Differentiated Services", RFC 2475, Dec 1998
19. Open Pluggable Edge Servies (OPES), http://www.ietf-opes.org/
20. Jeremy Elson and Alberto Cerpa, Editors, "ICAP, the Internet Content Adaptation Protocol", 2001
21. 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 04/02/01]
22. M. Hicks et al. PLANnet: An Active Internetwork. In Conf. on Computer Communications, INFOCOM'99, pages 1124–1133, New York, NY, Mar. 1999.
23. 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]
24. 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]

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

26. Jonathan M. Smith, Programmable Networks: Selected Challenges in Computer Networking, Computer, January 1999 (Vol. 32, No. 1), pp. 40-42.

27. Javed I. Khan & Yihua He, Fast Stream Intercept for high Performance Filter Appliances, Technical Report: 2000-03-01, Kent State University, [available at URL http://medianet.kent. edu/ technicalreports.html, also mirrored at http:// bristi.facnet.mcs.kent.edu/medianet]