Network Edge Services

Alfred C. Weaver, University of Virginia
Michael W. Condry, Intel Corp

SUMMARY

The Internet and the World Wide Web are widely viewed as operating on a client-server architecture. While this was initially considered ideal because it insulated the network from needing any knowledge of its traffic types, experience has shown that traffic type does matter. Much effort has now been expended to add data caches, media gateways, and proxy servers to make data appear more "local," and Content Delivery Networks have been invented to organize the vast array of Internet content on behalf of the client. In this paper we propose an alternative architecture that moves user interface issues away from the origin server and positions them at the network's edge, typically on the last hop that delivers data to the client. This so-called NetEdge appliance makes data transformations on demand (e.g., language translation, image transcoding, local content insertion) and accommodates both user preferences and the media adaptations needed to match a given user's capabilities. By distributing these tasks from the origin server to the various NetEdges located around the Internet, we simplify the user's interface, improve the quality and quantity of services provided, and foster a trusted environment in which vendors may add new services through a secure administrative interface.

Network Edge Services

Alfred C. Weaver
Department of Computer Science
University of Virginia
151 Engineer's Way
Charlottesville, VA 22904
weaver@virginia.edu

Michael W. Condry
Intel Labs
M/S JF3-206
2111 NE 25th Street
Hillsboro, OR 97124
condry@intel.com

NetEdge is a Network Edge device for the Internet that offloads the origin server by accommodating client capabilities and preferences locally. Services available include language translation, virus scanning, security and authentication, caching, image transcoding, and resolution of questions and ambiguity. We discuss how this new concept could be used as a virtual assistant to improve the client's view of available web services.

I. INTERNET EVOLUTION

The Internet's client-server architecture was adequate when data was primarily text, and client diversity meant distinguishing whether the client was connecting from a PC, MAC, or workstation. However, as the data to be delivered have become richer, and as the clients to whom it is delivered have become more diverse, Internet architecture has evolved to cope with the changes. Media gateways are used to interface wireless clients; caches are used to improve the delivery of static information; encryption enhances the privacy of data delivered; security and authentication modules attempt to protect the enterprise infrastructure from hackers and terrorists; and content delivery networks attempt to assemble dynamic content from databases, streaming servers, and content servers to improve performance by making accesses appear more "local".

But the proliferation of boxes that perform vendor-specific services is not an effective answer to a distributed "edge services" computing model for the Internet. A better approach is to design a common architecture for the specification and delivery of edge services. Service determination results from many factors including the request, client, and server, so the architecture logically has a programmable rule engine to identify and match "in the network" services. This allows the specification and implementation of future services, and packages everything into a NetEdge box (Fig. 1). Note that this approach does not require that all Internet users abandon the current Internet architecture or their current investments; rather, it offers an evolutionary growth path that is logical, programmable, and expandable so that it can accommodate future needs and changes.

NetEdge is not a repackaging of existing software; it is an evolution of the Internet with standard interfaces allowing for programmable technologies to be formulated and operate in a multi-vendor environment. It evolves these service technologies, such as Edge Side Includes [3], and creates a platform for the "in the network" aspects of web services using the Web Services Definition Language [4] (WSDL). Services have associated policies, security, and specifications as specified by the Intermediary Rule Markup Language [5].

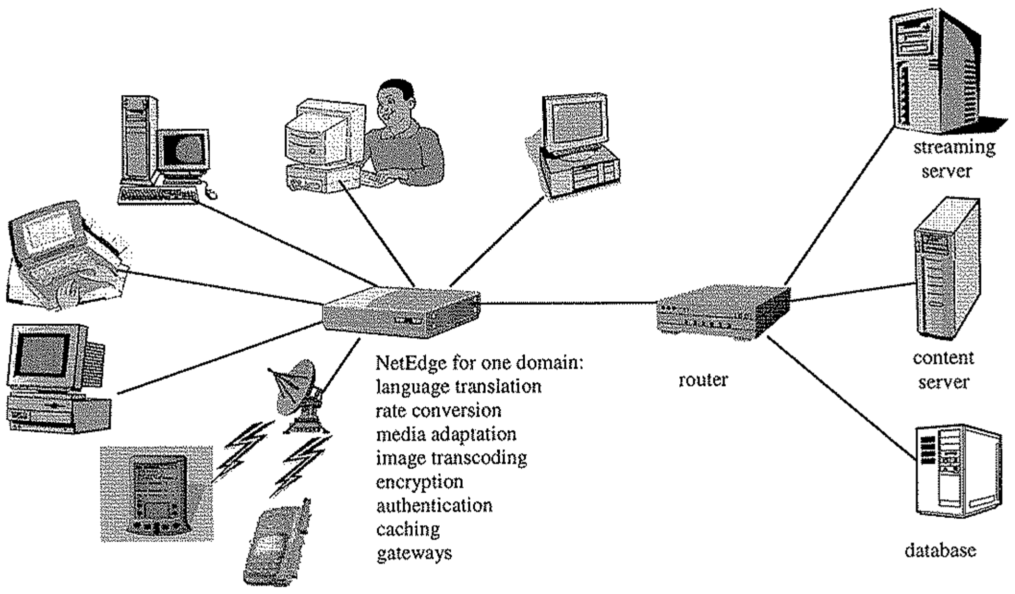

Fig. 1. Network Edge Services in the Internet

II. NETEDGE SERVICES

A. *Image Recoding*

In our teleradiology work with the U.Va. Health Sciences Center, we encountered a case that begs for a NetEdge solution [8], [9], [10]. The context is the collection, dissemination, and interpretation of medical diagnostic imagery. In our case the particular situation was that sonographic images (digital ultrasound) were being collected and stored in the industry standard DICOM (Digital Imaging and Communications for Medicine) format on the medical center's PACS (Picture Archiving and Communications System). For routine cases, the staff radiologist loads the DICOM playback software, selects a particular patient's DICOM image file, plays back the "movie" of the examination, and dictates an interpretation. That works well when all the data is local and can be exchanged via high-speed LANs (the raw imagery can be 2GB; a severely edited DICOM-encoded examination can be 100MB).

What happens when the radiologist is not co-located with the PACS? If access is achieved remotely over WANs (e.g., DSL, cable modems, telephone lines) the bandwidth is not there to shuttle large images, thus requiring image compression. But what kind of compression does the interpreting radiologist trust will not distort the image and thus reduce its diagnostic accuracy? Should it be motion JPEG? MPEG-2? MPEG-4? MPEG-7? wavelets? fractals?

An elegant solution would be for the NetEdge, using its profile of the consulting radiologist's image compression preferences, to transcode the DICOM image on-the-fly as it moves through the box (see Fig. 2). With this solution, the PACS retains only one version of the imagery (the original DICOM version), but the radiologist can dynamically choose which compression scheme best fits his or her own preferences and his/her viewing device's capabilities (e.g., resolution, contrast, bandwidth), and the imagery is transcoded enroute to fit those preferences and capabilities.

B. *E-Commerce Assistant*

Fig. 3 illustrates an e-commerce example. Suppose a user has voluntarily provided his or her profile for storage on the NetEdge. If the user's profile contains two credit card descriptions, say one personal and one corporate card, and if the user's purchase information is submitted via an XML-based form, it is typically the responsibility of the origin server to determine which credit card to use. If the user is ordering from a desktop browser, this is easily resolved by displaying an HTML form with two radio button selections, and the user chooses one or the other. But what if the user is ordering from a shopping kiosk with a touchscreen? Or a PDA? Or a cell phone? Then the origin server must anticipate each potential mode of interaction and provide interfaces for each. Worse, these services must be repeated by every store supported by every server.

C. *Natural Language Translation*

Although natural language translation by machine does not yet have the quality of human translation, it has reached a certain level of utility. In the area of e-commerce, for example, one can envision a shopping assistant that translates among common languages at a level of proficiency that allows a native English speaker to shop in a French language e-store (but probably not at a level sufficient to translate novels or convert legal contracts or give medical advice). These translation services are currently available from sites such as babelfish.com [6] on a manual, page-by-page basis, and our proposal here is that the NetEdge service automates and regularizes their invocation through a rule language. For example, when the native web page is in German and the viewer has expressed a preference for English, the NetEdge service invokes the German-to-English translator on the text, and the customized output is then delivered to the viewer in English.

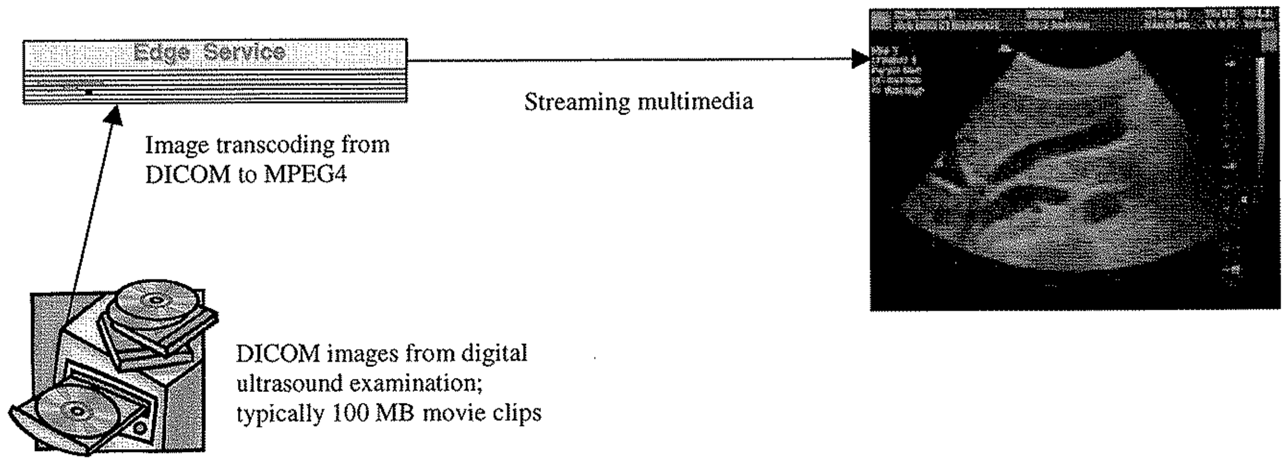
Edge Service
Streaming multimedia
Image transcoding from DICOM to MPEG4
DICOM images from digital ultrasound examination; typically 100 MB movie clips

Fig. 2. Medical Image Transcoding

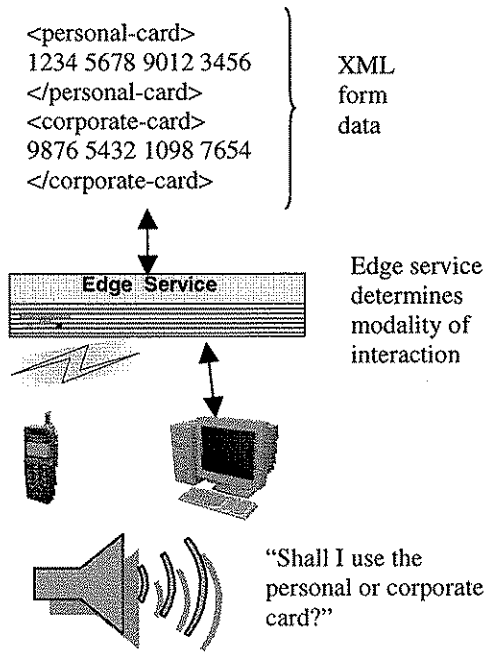

Fig. 3. Media Adaptation (text to voice)

We have used this example with eBay shopping and the results are encouraging. In our experiment, the information source is an eBay page listing the results of a search for DVDs; the native display is in English. By successively invoking the English-to-French and English-to-German translation rules, the NetEdge (using callouts to translation services) can produce translations of sufficient quality to use this particular e-commerce application (see Fig. 4).

III. NETEDGE ARCHITECTURE

NetEdge architecture focuses on extending the proxy-like "edge" device to be an "in the path" device. As we indicated, the services of interest are those that transform data rather than "own" it. Content Delivery Networks follow this model, organizing and presenting content according to the wishes of the content provider. Other services (ISVs, corporate gateways) provide customized content according to the preferences of the requestor (client). Our project builds a platform to architect services that augment or transform request/reply information between the client and content provider(s).

The edge appliance determines the service that is required and, typically, the service is executed with an "out of path" or "callout" computation engine. The objective of the NetEdge design is scalability and flexibility, particularly in how services can be invoked. The scalability focus is concerned with not overloading "in the path" agents, and the flexibility focus is concerned with executing services in different operating environments. To this end, the NetEdge appliance runs services with a multi-tiered approach as shown in Fig. 5.

As its core, the NetEdge appliance works as a proxy device, either as a forward proxy (e.g., in an ISV) or as a reverse proxy (e.g., in a CDN), providing an open platform for performing services on the content passing through it.

A Rule Engine in the edge device determines which services are needed for a given request/reply sequence (e.g., a client can choose an optional translation based on the country indicator in the URL and the browser's language setting). This eliminates evaluating network traffic that does not require services; today that may be a majority of the network traffic, so without a match the traffic passes through untouched, thereby maximizing performance. In addition, when a service is required, the Rule Engine provides exactly the service that is needed. The connection between the rule engine and the available services is done with a code fragment called a *proxylet* that is set up to invoke a particular service under a particular set of circumstances.

The Proxylet Interface has two available APIs, one in Java and one in C/C++. If desired, a service implementer can write directly to one of these APIs and have the service run directly on the local machine. However, since offloading the actual work of the service onto remote callout servers is often desirable, proxylets for remote callout protocols are also provided.

The three client proxylets already in use are RMI (Remote Method Invocation) and BEEP [12] (Blocks Extensible Exchange Protocol) in Java and ICAP [13] (Internet Content Adaptation Protocol) in C/C++. The language interface is chosen simply by what is most convenient to the application at hand.

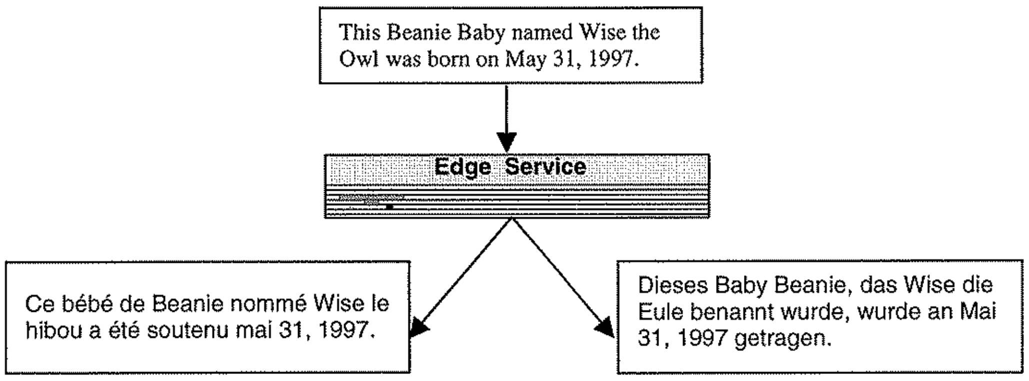
This Beanie Baby named Wise the Owl was born on May 31, 1997.
Edge Service
Ce bébé de Beanie nommé Wise le hibou a été soutenu mai 31, 1997.
Dieses Baby Beanie, das Wise die Eule benannt wurde, wurde an Mai 31, 1997 getragen.

Fig. 4. NetEdge Language Translation Service

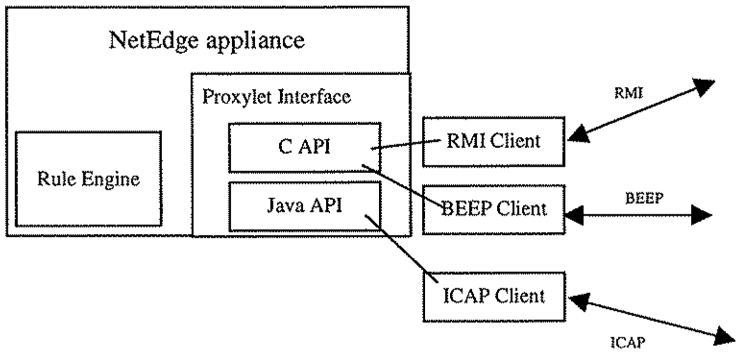

Fig. 5. NetEdge Architecture

The RMI interface exposes a well-defined API for modifying content; the RMI interface parallels the C and Java APIs for proxylets. The two adaptation protocols provided are ICAP and BEEP; both of these are application layer protocols designed for specific uses. ICAP, in particular, is designed to modify HTTP requests and replies and is therefore very useful for web traffic.

One of the benefits of this multi-layered approach is the ease of supporting new protocols; for instance, the support of SOAP [14] or Corba could be added by simply creating a Java or C proxylet to handle the new interface. A developer experienced with the NetEdge appliance and the new protocol should be able to add support in a matter of a few weeks.

At this point this architecture is an ad-hoc extension because today's Internet is populated with proxies (and reverse proxies) that serve as both a potential computational and caching agent for "in the network" services.

The Callout Service Engine(s) where nearly all the data transformations are performed as an "in the net" service is any compute engine outside the basic network path. In a CDN or ISP world, this would typically be another compute engine in the rack of systems used for web services.

The NetEdge box communicates with this engine by using a standard protocol to request the service. Some environments assume control of the data traffic that is being sent back to the client and others use the NetEdge box. This is defined with the interface proxylet and control protocol between NetEdge and the Callout.

Service administration would be accomplished as shown in Fig. 6. Each service provider would interface to the service administration SDK using SOAP. Each service vendor and rule owner would adhere to security policies (e.g., XML digital signatures) as defined by the IETF. The service administration SDK would provide an administrative user interface, security checking, and operate a rule parser.

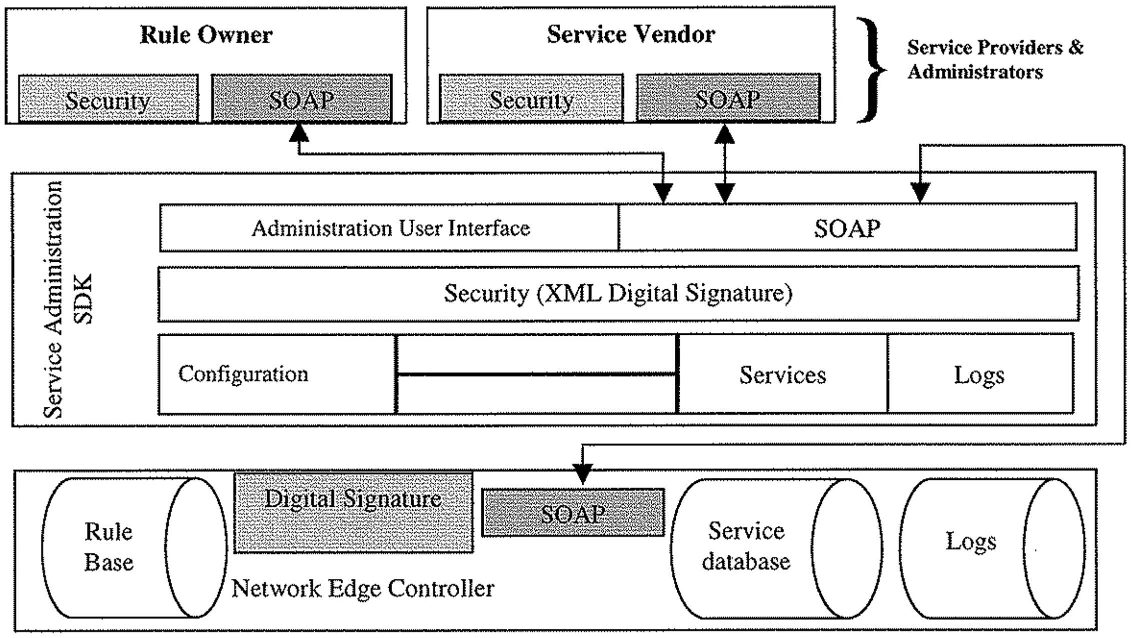

Fig. 6. Service Administration

All of the above would then interface to the network edge controller (rule base, service logs, and service database) via SOAP.

IV. SUMMARY

OPES services, implemented via the NetEdge, expand the client's access to a variety of data streams. Of particular benefit is the fact that origin servers continue to "own" their content as they do now; NetEdge boxes perform data transformations as data moves through them enroute to the client. The NetEdge accommodates the user's preferences and capabilities and media, rather than requiring every origin server to anticipate every access device and every client preference. By distributing computation to the network's edge, the origin server is simplified, the original data is kept in only one place (easing version control), and the user's preferences are satisfied based upon the NetEdge's permissive knowledge of the client's wants and needs.

V. REFERENCES

[1] Akamai EdgeSuite, *http://www.akamai.com*

[2] Internet Engineering Task Force, Open Pluggable Edge Services, *http://www.ietf-opes.org/*

[3] Edge Side Includes, *http://www.esi.org /dload /esi_overview.pdf*

[4] Web Service Definition Language, *http://www.w3.org/TR/wsdl*

[5] Intermediary Rule Markup Language, *http://www.ietf-opes.org/documents/draft-beck-opes-irml-00.txt*

[6] Babelfish language translators, *http://wwww.babelfish.com*

[7] Extensible Markup Language, *http://www.w3.org/XML/*

[8] Sublett, J.W., Dempsey, B.J., and Weaver, A. C., "Design and Implementation of a Digital Teleultrasound System for Real-Time Remote Diagnosis," Computer-Based Medical Systems, IEEE Computer Society Press, 1995, pp. 292-298.

[9] DeAngelis, Gia A., Dempsey, Bert J., Berr, Stuart, Fajardo, Laurie L., Sublett, John W., Hillman, Bruce J., Weaver, Alfred C., Berbaum, Kevin, and Dwyer, Samuel J., "Diagnostic Efficacy of Compressed Digitized Real-time Sonography of Uterine Fibroids, " *Academic Radiology*, Volume 4, 1997, pp. 83-89.

[10] DeAngelis, Gia A., Dempsey, Bert J., Berr, Stuart, Fajardo, Laurie L., Sublett, John W., Hillman, Bruce J., Weaver, Alfred C., Berbaum, Kevin, and Dwyer, Samuel J., "Digitized Real-Time Ultrasound: Signal Compression Experiment," *Radiology*, Vol. 197(P), November 1995, p. 336.

[11] Weaver, A.C. and Van Dyke, J., Multicast Distribution and Control for Streaming Multimedia, Proc. IECON'01, December 2001.

[12] BEEP: Blocks Extensible Exchange Protocol, *http://www.ietf.org/html.charters/beep-charter.html*

[13] ICAP: Internet Content Adaptation Protocol, *http://www.i-cap.org/*

[14] Simple Object Access Protocol, version 1.1, *http://www.w3.org/TR/SOAP/*