Media Companion: Delivering Content-oriented Web Services to Internet Media

Wei-Ying Ma, Chun Yuan, Xing Xie, Yu Chen, Liang Sun, Wanghong Yuan
Microsoft Research Asia
3F, Sigma Center, No 49 Zhichun Road
Beijing 100080, China
{wyma, i-cyuan, i-xingx, i-yuchen}@microsoft.com

ABSTRACT

In the past few years we have seen a huge industrial investment on the development of content delivery networks (CDNs) which provide a large number of caches and storage devices at the edge of network to push content closer to the user for fast and reliable delivery. This development has triggered the beginning of a so-called "edge computing" where applications and computational resources are also moving to the edge for intelligent media services. To cope with the growing diversity and heterogeneity of the Internet, these edge servers offer a natural place to extend the capability of network intermediaries for content-oriented services such as automatic adaptation for small devices, personalization, location-aware data insertion and virus scanning.

In this paper we present our investigation on using a network of edge servers for delivering content-oriented web services to Internet media. Similar to CDNs, our design goal is to make the content-oriented services part of the Internet infrastructure services accessible to content providers and content consumers via a subscription model. A prototype system called "Media Companion" which is built to evaluate our design principles is described in the paper, and the experimental results of deploying this system in our corporate network are also discussed.

Keywords

Content delivery network, web services, adaptive content delivery, personalized content, multimedia, media proxy

1. INTRODUCTION

As the Internet is moving toward the service-centric model, more and more storages and computational resources are being put into the network infrastructure and provided as services to customers. In the content networking world, for instance, this trend can be seen on the development of content delivery networks (CDNs) [1][10][25] which make content distribution a network infrastructure service available to content providers and network access providers. Furthermore, the progress on standardization and development of Web services has marked the beginning of a new era that every computational resource and service on the Internet can be connected to provide new user experiences on accessing, sharing, and using information anytime, anywhere, from any device. The Microsoft's .NET initiative has been aiming to realize this vision by providing the platform and foundation Web services. The .NET My Services [23], previously codenamed Hailstorm, is being developed to provide a set of user-centric XML Web services oriented around people. For example, services such as myInbox, myCalendar, myContacts, myAddress,

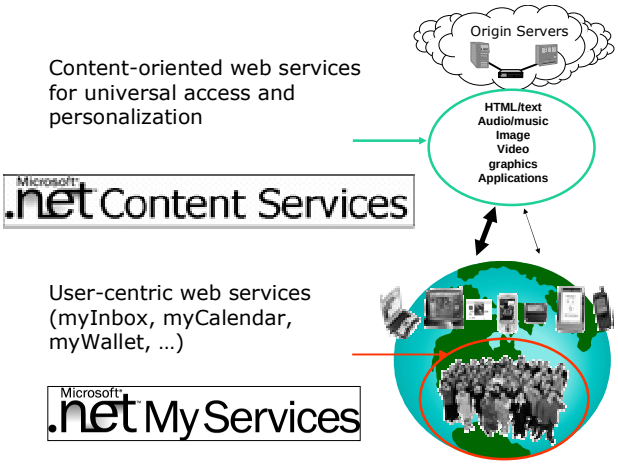


Figure 1. The user-centric web services help users manage their information across all the devices while the content-oriented web services process content and media objects to enable universal access and personal experience from any devices.

myProfile, myWallet, myLocation, and myNotifications will be provided to people to help manage their information and interactions across all the applications, devices, and services in their lives. These foundation Web services use standard communication interfaces and protocols based on XML, SOAP [15], UDDI [34], and WSDL [9] so that other service providers could use them to compose and develop new Web services for people.

While the user-centric Web services are being addressed, there still remains the problem of making sure that content (or media) on the Internet can be adapted according to network environment for universal access. Also, other content-oriented processing such as personalization, customization, local content injection, watermarking, data hiding, and virus scanning are becoming more and more important while the content flows from one end to the other end of the Internet. It would be extremely useful if such content-oriented processing could also be made as Web infrastructure services to process the content on behalf of people and content providers. Figure 1 illustrates our ultimate goal of using both user-centric and content-oriented Web services to provide users pervasive and personal media experiences.

The content-oriented Web services will require a very different mechanism to deploy in comparison to traditional Web services, as they need to be injected into the current content delivery path. A simple distinction is that these content-oriented services will be

enabled in network intermediaries such as proxy caches and delegates while traditional Web services are essentially server services. As of today, such a Web middleware infrastructure for content-oriented services is still in its infancy, and there are many open research issues need to be further investigated.

Having realized the importance of content-oriented services on the Internet, the current CDN providers are increasingly looking into upgrading their current delivery networks to enable such value-added services. In our previous work [21], we have proposed a system architecture and protocols for building such intelligent CDNs. In this paper, we describe our latest progress on further refining and developing such a system. In particular, we focus on the mechanism of binding the subscribed services with the content to instruct edge servers to perform necessary actions. Our goal is to make the content-oriented Web services an infrastructure service accessible to customers that could be content providers, end users, ISPs, or traditional CDN providers. This service agreement is uploaded to the edge servers and instructs them to perform services while data is flowing through them across the Internet. To achieve this goal, we also ensure that the service model interacts with other existing network elements collaboratively and seamlessly so as not to undermine the success of end-to-end nature of Internet client/server interactions.

Here we summarize the novel contributions of our work:

1. The idea of enabling a large scale of content-oriented processing on the Internet as subscription-based Web middleware services is first introduced in the literature.
2. A framework which defines the required mechanisms and system interactions between various network entities to accomplish the task is proposed.
3. A first working prototype called Media Companion has been developed to demonstrate our vision and usefulness of the system. Five major content-oriented services have been developed and are available for subscription and real usage.

This paper is organized as follows: We discuss the framework of content services network and its two major parts, i.e. service delivery overlay and service-enabled proxy in Section 2. Section 3 talks about how to deliver content-oriented service in the network intermediaries using a subscription model. Section 4 describes our prototype system and several developed content services. The related works are discussed in Section 5. Section 6 summarizes our work and presents our future directions.

2. CONTENT SERVICES NETWORK

In content delivery networks (CDNs), a large number of caching proxies is deployed at the network edge and used as a distribution channel to push content closer to the end user. The content is replicated to a set of edge servers based on estimated demand and a caching policy, and the DNS-based redirection technology is used to ensure the network and server load are balanced. It also ensures the availability of the content by making multiple copies within the distribution network. As the current CDNs only move content and makes sure it is delivered reliably and promptly to end-users, they do not modify the content by adding value to it.

To inject content-oriented services into the current content delivery path, the caching proxies of content delivery works need to include a certain new capabilities. A new network infrastructure which constitutes of an overlay network of application servers for deploying and replicating the content-oriented services also need to be in place. Our design principle is to separate the roles of storage (caching) and computation (processing and transcoding) in the network infrastructure. As traditional Web caches have been specially designed to perform efficient hashing for finding if the requested content is in local storage and then quickly respond to the request, putting additional processing burden in web caches will compromise the caching performance. Therefore, we propose two layers of network infrastructures, which keep the original content distribution layer (i.e., CDNs) as it is in providing web caching service while another layer offers computational resources to deliver value-added services in the content delivery path (see Figure 2).

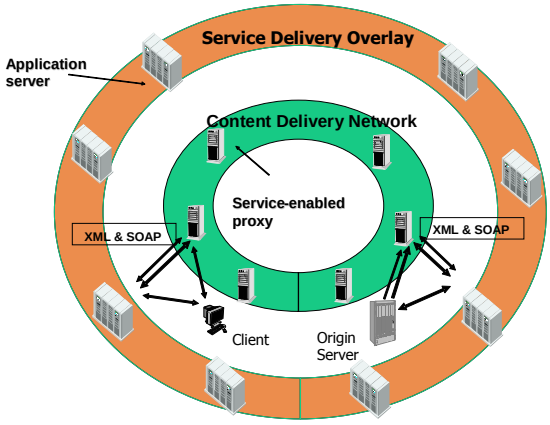


Figure 2. The content services network (CSN) consists of two separate overlay networks. In the content delivery layer, service-enabled proxies are needed in order to interact and collaborate with application edge servers in the service delivery layer. SOAP and XML message are used in the communication protocols between these two layers to inject content-oriented services in the content delivery path.

These two overlays together create the so-called content services network (CSN), which will provide intelligent content-oriented services while moving the content on the Internet. One thing to note is that there are mainly two types of edge servers in the CSN framework: one is the service-enabled proxy of the content delivery layer which extends the functions of a traditional web cache for performing value-added services; the other is the application server of the service delivery layer which acts as a remote call-out server for the first.

The amount of new work need to be done in the service-enabled proxy should be minimized to mostly checking if an additional processing or special handling is necessary for a given content. The heavy-weight processing is conducted by an application server via a remote call-out protocol (e.g. SOAP). We will discuss the additional functionalities which need to be equipped in the service-enabled proxy in Section 2.2. And service binding and service execution which make the two layers collaborate together will be discussed in Section 3.

2.1 Service delivery overlay

The service delivery overlay provides the distribution channel to push applications (media processing and transcoding) to the edge.

As shown in Figure 3, it consists of the following three major elements:

Application servers: host the software of content-oriented processing for content delivery. These application servers provide computational resources to process content on behalf of content providers or content consumers.

Service delivery and management (SDM) servers: are responsible for the following tasks: (a) they register and publish the content services provided by service providers. (b) They distribute the application within the service delivery overlay by replicating the corresponding services to a set of edge servers based on the estimated demand. (c) After initial service registration, publication and distribution, SDM servers continue to monitor the performance of each service and dynamically adapt the scale of distribution and deployment of the service according to its demand. (d) SDM servers aggregate the information about usage of services and then provide it back to the redirection servers. The redirection servers use this information to perform network and server load balancing. (e) SDM server also provides management, accounting and billing functionalities to service providers. (f) Each SDM server is responsible for collecting information about its domain and periodically exchanges the information with other cooperating SDM servers.

Redirection servers: Direct service request to an application server according to a number of attributes and measurements. It receives information from SDM servers.

The application servers and SDM servers communicate with the caching proxies in the CDN layer to complete the service subscription and rendering. We will discuss the detailed system interactions in Section 3.

2.2 Service-enabled proxy

The service-enabled proxy is the key component to enable content delivery layer to interact with the service delivery layer in the CSN framework. Figure 4 shows the framework of service-enabled proxy, which consists of six main parts: (1) an instruction parser, (2) a message parser, (3) an instruction processor, (4) a service execution module, (5) an instruction cache and (6) a result cache. We build our proxy prototype using Microsoft ISA Server 2000 [24]. The detail of our implementation will be discussed in Section 4.1.

Below we describe the basic operations performed in the service-enabled proxy:

Service provision:

(a) Instructions are transferred from the SDM servers to service-enabled proxy. These instructions represent service subscription information from content providers or end users.

(b) The instruction parser compiles the instructions received from the SDM servers and stores it in a local instruction cache.

Operations in the run time:

(1) A registered type of messages defined in the instruction will trigger the proxy to activate the message parser which will then parse current message (HTTP request or response) to extract necessary information such as content type or language.

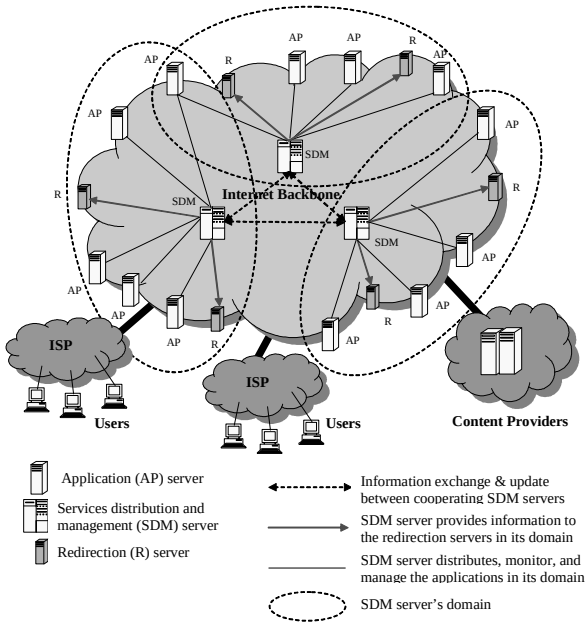


Figure 3. The system overview of service delivery overlay to push content-oriented services to the edge of network

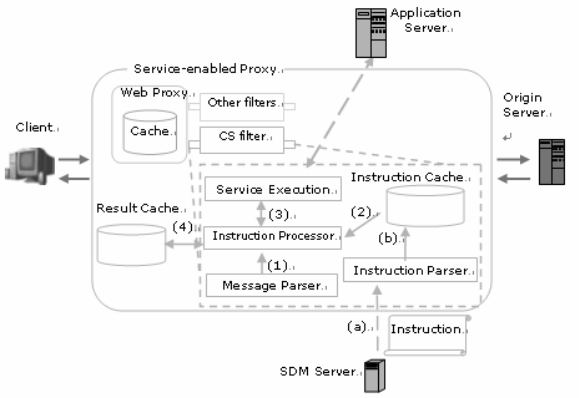


Figure 4. Framework of a service-enabled proxy

(2) The instruction processor loads related instructions from the cache. It either calls a local service execution module or invokes a remote service.

(3) If necessary, service execution module will be employed to invoke a remote service in the application servers through the SOAP protocol.

(4) After processing, the results will be saved in a result cache for future use. At the moment, we only consider serving the saved result to the original service subscriber due to privacy issue.

Note that the local service execution module is only meant to perform light-weight processing. For these local services, there is a need to define a standard runtime environment in the proxy so that a common platform can be used to develop new applications. The IETF working group on OPES [27] has been formed to define such a standard. Once the behavior of future service-enabled proxy can be agreed upon, it will facilitate the integration of web services and content networking worlds.

2.3 Using UDDI for Service Registration, Publication, and Discovery

Universal Description and Discovery Integration (UDDI) [34] is an open and global registry which allows businesses to register information about their Web services so that other businesses can find them. CSN uses UDDI to publish content-oriented services for public discovery and access. Service subscription/un-subscription, query and configuration interfaces are registered in UDDI so that end users or content providers who are interested in a particular service may send the requests to CSN according to service type specifications found in UDDI.

3. DELIVERING CONTENT-ORIENTED SERVICES VIA A SUBSCRIPTION MODEL

Several steps are involved to deliver content-oriented services at the edge of network. The first step is to register and deploy content-oriented services in the edge servers, which include application servers in the service delivery layer and the service-enabled proxy in the content delivery layer. The second step is for service subscribers to specify what services need to be bound to their domains, media objects, or user identities. The services are invoked by edge servers during the run time according to the service instructions.

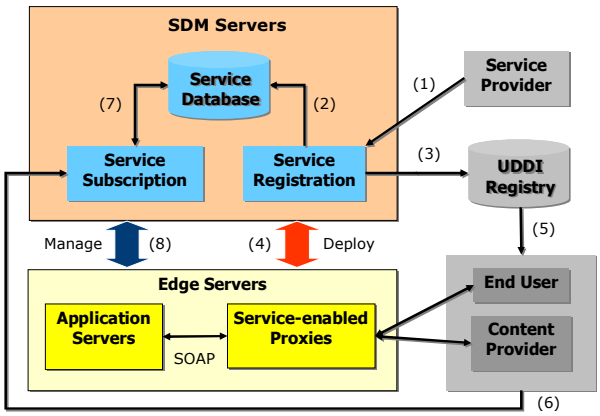
3.1 Service registration and deployment

As we have discussed previously, the content services network provides the necessary network infrastructure to distribute and deploy content-oriented web services in a scalable and fault-tolerant manner. Before the service becomes available, it needs to be registered in the UDDI registry first. CSN receives components such as service specifications and binaries from service providers and stores them in the database. It then publishes service information to the UDDI for public discovery and access.

Along with the service registration and publication, the corresponding application or executable module is provided to the SDM server of service delivery overlay, which then dynamically selects a number of edge servers to deploy the service based on estimated demand and required geographical coverage.

There are two types of edge servers in CSN to host applications and executable modules. The light-weight services can be provided as executable modules uploaded to service-enabled proxies and executed in the content delivery path. The heavy-weight services are installed in the application servers and invoked by the service-enabled proxies through the SOAP protocol as described in Section 2.2.

For each registered service, a service specification should be provided to describe invocation interface and execution environment of the service, such as parameter types and transport protocols that the service operates on, and the associated conditions (such as content type, user preference, device capability, location, and network bandwidth) to invoke the service in the edge servers. Binaries include applications running on application servers and executable modules running on service-enabled proxies. Figure 5(a) shows the system interaction between different components in CSN. Figure 5(b) describes the process of service registration of deployment.



(a)

Service registration and deployment:

(1) The service provider provides its content-oriented service
(2) The corresponding service and the service specification are stored in the service database
(3) The service is registered in UDDI registry for public discovery and access
(4) The service is deployed and enabled in edge servers

(b)

Service subscription and binding:

(5) The subscriber finds the service from UDDI registry
(6) The subscriber sends the subscription request to the SDM server
(7) The subscription service stores the subscriber-service bindings in the service database
(8) Service instructions representing subscriber-service bindings are transferred to service-enabled proxies

(c)

Figure 5. (a) The system interaction between various components in the CSN framework. (b) The process of service registration and deployment. (c) The process of service subscription and binding.

3.2 Service subscription and binding

One fundamental requirement to enable content-oriented Web intermediary services is determining whether current content (e.g. HTTP messages) should be serviced and how to invoke the service (service instruction). Because service contracts are between service providers and communication ends such as origin content servers and clients, there needs a way to make the edge servers aware of the binding between the content requested or delivered by the subscriber and the subscribed services.

There are several possible solutions to service subscription and binding. A straightforward approach is to label the content (e.g. HTTP messages) to indicate that it needs special handling. Proxies that intercept the labeled content will take appropriate actions according to the instruction. For content labeling, there are two choices: 1) embed service instructions directly into content (e.g. HTTP headers), or 2) attach an indicator (URI) to the

content which refers to an external document that describes service instructions. However, both methods require the change on origin content server (if the subscriber is a content provider) and user agent in client device (if the subscriber is an end user), a significant obstacle for CSN deployment.

Another approach is to let the subscriber and the service provider determine on the service contract, and then let CSN translate that contract into a service binding which describe the association between the subscribed services and the end user or the domain name/media objects (if the subscriber is a content provider). Service bindings are maintained by CSN which knows the current status of subscription and constantly updates service-enabled proxies with the latest service instructions (see next section). The subscriber or the service provider only needs to interact with CSN, which allows the service delivered transparently in the existing infrastructure.

Content providers and content consumers find their interested services in the UDDI registry first, and then follow the access point to CSN's service subscription interface. Once the subscriber confirms the subscription of the service and related configuration, CSN will store the binding in the database. The binding enables the future delivery of the subscribed services to the content. It also contains the conditions on which the service should be invoked. The following are examples of service binding. Figure 5(c) illustrates the process of service subscription and binding.

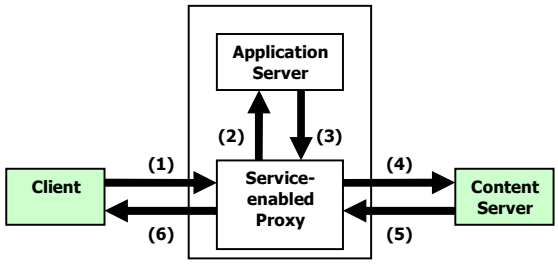
```
<binding type="end user">
  <user id="R&$%GHJU"/>
  <service name="web page adaptation">
    <condition>
        <condition name="user-agent" matches="Pocket PC"/>
        <operator name="and"/>
        <condition name="MIME-type" matches="text/html"/>
    </condition>
  </service>
</binding>

<binding type="content provider">
  <domain name="www.microsoft.com"/>
  <service name="watermarking">
    <condition>
      <condition name="MIME-type" matches="image/jpeg"/>
      <operator name="and"/>
      <condition name="request-path" matches="/gallery/*"/>
    </condition>
  </service>
</binding>
```

3.3 Service enabling and execution

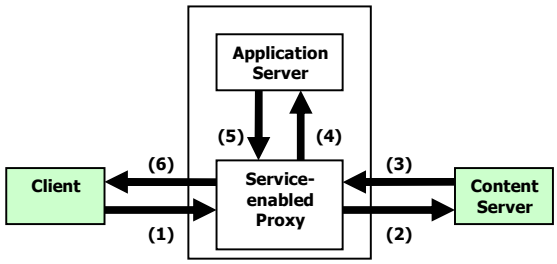
A service instruction represents a binding to a single service and contains the actual service access point. The instructions will be transferred from SDM servers to the service-enabled proxies that the subscriber is associated with so that it will act according to the instructions to execute services for the subscriber at appropriate moments. The transfer may be eager (SDM server sends the instructions to the proxy immediately after the subscription) or lazy (the proxy fetches the instructions from CSN for the subscriber on a periodical or per-session basis). The proxy has a cache/replica to store service instructions representing a subset of service bindings for its domain and keeps them up-to-date.

The proxy verifies whether the message needs additional services according to the instructions. If the message and its related context satisfy the condition specified in the instruction to execute a service, the proxy will provide the service locally or pass the message and related parameters to the application servers. The processed result is returned to the proxy and then goes to its original destination.



(1) The client sends a request message
(2) The service-enabled proxy checks the message and detects it needs additional service; it forwards the message to a remote application server or invokes a local executable module
(3) The application server returns the result to the proxy
(4) The proxy forwards the modified request (assuming the processing result allows so)
(5) The content server returns a response message
(6) The proxy forwards the response to the client.

Figure 6. Service executed on client requests.



(1) The client sends a request message
(2) The service-enabled proxy forwards the request to the content server
(3) The content server returns a response message
(4) The proxy checks if the message needs additional service; if so it forwards the message to a remote application server or invokes a local executable module
(5) The application server returns the result to the proxy
(6) The result is delivered to the client

Figure 7. Service executed on server responses.

A service may be executed on the client request or the server response of a session. Request filtering and security adaptation [14] belong to the former type, while many other services belong to the latter type, such as Web page adaptation, watermarking, virus scanning. Figure 6 and 7 depicts the service execution in these two cases.

4. Media Companion: The Prototype System

We have built a prototype system called Media Companion to evaluate our system design. This system is currently deployed in the corporate network of Microsoft Research Asia with five example services available for subscription. We describe the details of our implementation in the following.

4.1 Implementing a service-enabled proxy

As we have mentioned previously, the service-enabled proxy is the key component of CSN. We build a prototype of such proxy based on Microsoft Internet Security and Acceleration (ISA) Server 2000 [24].

Microsoft ISA Server 2000 is an extensible firewall and Web cache server. It provides Internet Server API (ISAPI, previously available with Microsoft International Information Server) that can be used to develop users' own extensions named Web filters which may provide functions like virus scanning, compression, logging or data encryption. Web filters allow you to intervene in the processing of HTTP requests. They are dynamic-link libraries that are loaded when the ISA Server is started and stay in memory until the service shuts down. Web filters can be configured to receive special filter-event notifications that occur with each HTTP request that the proxy receives and with each response returned through proxy.

We use the HTTP port to deliver the service instructions from SDM servers to this service-enabled proxy. If the proxy receives a message whose first line is "CSN Instructions", then it will read the following lines as an "instruction module" and save it into a local cache. In our prototype, an instruction module contains instructions for a single content provider or a single content consumer. A newly uploaded module will completely replace the old one. An instruction module typically contains several instructions for service binding.

We developed a Web filter named content service filter (CS filter) using ISAPI which enables additional processing on HTTP messages. After installing the CS filter, the ISA Server is turned into a service-enabled proxy. It will analyze HTTP messages (either request or response) passing by and performs necessary processing according to predefined instructions. These instructions are received from the SDM servers and contain service-binding information. The content processing may be executed locally on the proxy server or by another application server through a remote callout interface. In our prototype, we use SOAP Toolkit 2.0 [31] to implement the remote callout interface.

The identification of users is useful in many content services like personalization. In our implementation, we use authentication mechanism of the ISA Server to identify different clients, and therefore each client is required to log on to the proxy before using the personalization service. This method enables a more precise way of identifying individual user than judging from the IP address.

4.2 Service subscription interface

The Media Companion uses Microsoft Visual Studio .NET for developing service binding and enabling. The system consists of a Web interface written in ASP.NET and a middle layer written in C# which parses Web requests from users and accesses the service database which is built using Microsoft Access. The database is

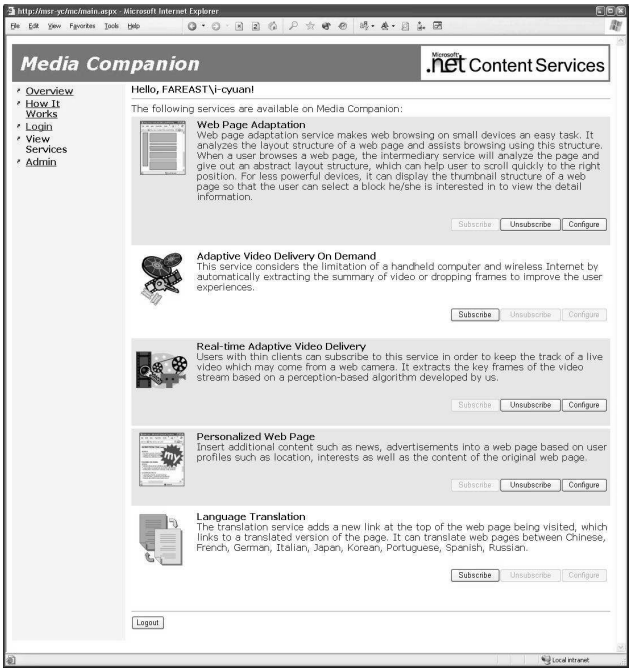


Figure 8: The service subscription interface of Media Companion. Five content-oriented services are available for subscription.

used to store binding relationships between subscribers and services. It also contains service-related information such as execution conditions, parameters and location. Figure 8 shows the main service subscription interface of Media Companion.

When a subscriber logons to Media Companion, it must first identify itself as an end user or content provider since the identification will determine the type of instruction which will affect the proxy behavior in message parsing and preparing for subsequent service execution. If he/she is an end user, the identity should also be provided to Media Companion. Service-enabled proxies will use this identity to execute services for different user. Our implementation uses Integrated Windows authentication to obtain the identity of the end user automatically. The same identity will also be obtained by ISA proxy server when the user requests content through it and the proxy will activate the services he/she has subscribed to.

If he/she is a content provider, it is required to identify the domain name where the content is provided and accessed. The domain name will be used by proxies to execute services for content from the corresponding domain. After the identification, Media Companion queries the service database to get the subscriber's subscription records and shows the list of available services for subscription/unsubscription and current configuration.

The subscriber may decide to subscribe to a new service or unsubscribe to a service he/she has subscribed to. When Media Companion receives a subscription request, it adds the new subscriber-service binding record to the service database. After reading the bindings related to the subscriber and the information about the services, it composes the service instruction and transfers it to the service-enabled proxy where the end user is

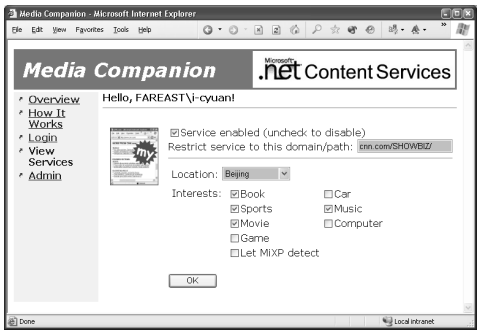


Figure 9: Media Companion provides an interface for subscribers to configure the services in a finer granularity.

associated. The proxy will update its cache with the new instruction for the subscriber. The unsubscription process follows similar steps.

The subscriber can also configure a service he/she has subscribed to. When Media Companion receives such a request, it presents the interface showing his/her old configuration (see Figure 9). The subscriber can modify the settings. After the modification is submitted, Media Companion refreshes the corresponding configuration stored in the service database and also composes a new service instruction and transfers it to the service-enabled proxy to change the service invocation accordingly.

For extensibility reasons, each service (if it needs configuration to work) is required to provide a component implementing an interface defined by our Media Companion system. The component is responsible for generating the Web interface of service configuration, restoring old settings and transforming user submissions back to settings which are recognizable by Media Companion.

4.3 Developed services

We have developed five different content-oriented services available in our corporate network. Each of them is described in the following.

4.3.1 Webpage adaptation for Pocket PC

The explosive growth of small Internet devices such as handheld computers, personal digital assistants (PDAs) and smart phones have been used to leverage the capabilities of the Internet and provide users ubiquitous access to information than ever before. Despite the proliferation of these devices, their usage for accessing today's Internet is still largely constraint by their small form factors such as small screen and limited input capabilities. Particularly, most today's web content has been designed with desktop computers in mind, and often contains large web pages which do not fit into the small screens of these devices.

Although some web sites and portals have started to provide special version of content tailored for wireless and mobile devices, the mobile user often encountered the need to browse traditional Web sites as they are still their primary information sources. Browsing a traditional web page in a small device like Pocket PC is like seeing a mountain in a distance from a telescope. It requires the user to manually scroll the small window horizontally and vertically to find and position the view correctly for reading

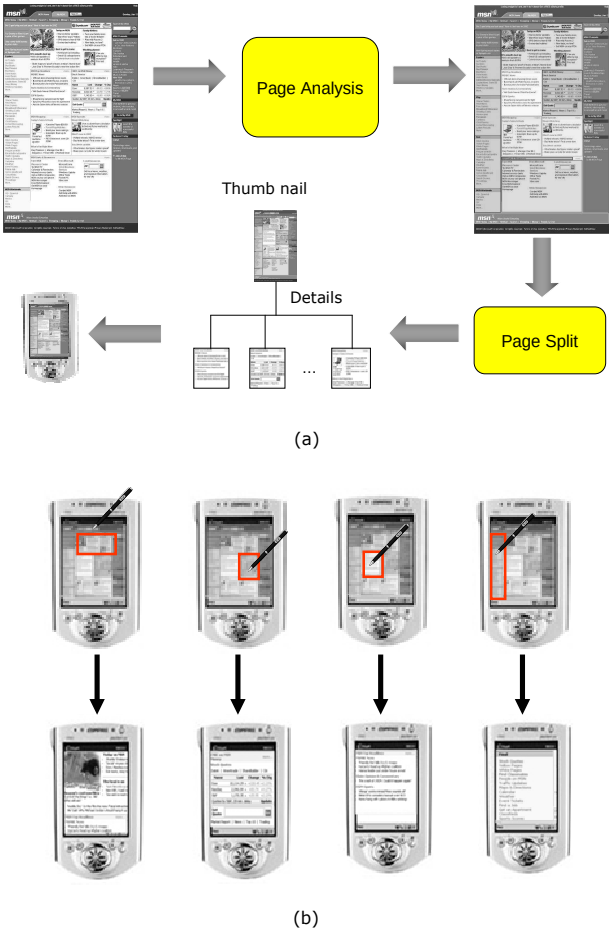


Figure 10. (a) Web page analysis to extract representation structure. (b) Efficient web browsing and navigation on Pocket PC.

information. This tedious and time-consuming browsing procedure has largely limited the usefulness of these devices.

To resolve this problem, we have developed a bi-level web page representation which allows a large web page to stay as a unit and preserve its original looking but with new zoomable and auto-positioning capabilities. This representation makes the browsing of a large web page in a small device an easy task. Based on this page representation scheme, we also developed an automatic web page analysis algorithm which analyzes the content and structure of a given web page and then generate a corresponding bi-level web page representation.

Figure 10(a) shows the process of our automatic page analysis. The algorithm utilizes the content, HTML tags and structure of a web page and then partitions it into a number of blocks which best fit into a targeted size of screen. The original web page is split into two levels. The top level contains a thumbnail representation of the page and the bottom level consists of all the small blocks as individually viewable pages. Each block is color-coded in the thumbnail to indicate it is individually zoomable. So the result is a two-step browsing process. When a user requests a web page using a small-screen device, the thumbnail is first presented to him for quick overview and easy navigation. Then he can decide to go to view a particular block of information by clicking the

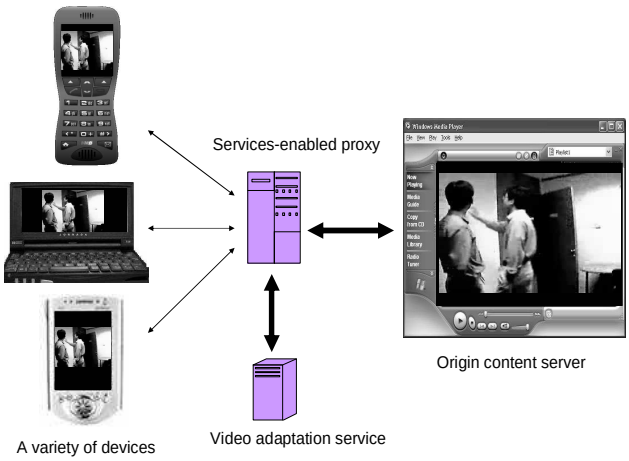


Figure 11. Adaptive video delivery to a variety of mobile devices.

corresponding region in the thumbnail. Figure 10(b) shows an example of browsing the MSN homepage on Pocket PC.

This webpage adaptation is made as a subscription-based service in our Media Companion prototype system. Though it is currently only available within our corporate network, our ultimate goal is make this service easily accessible to any content providers and end users to facilitate information delivery to the wireless Internet.

4.3.2 Adaptive video delivery

There has been much research work on delivering a video to the wireless Internet. The technical challenges include dealing with the limitation of bandwidth, signal fading and packet loss in the communication channel, and power consumption. In our Media Companion system, we deploy a simple adaptive video delivery scheme as a subscription-based service to the mobile user. Before streaming the entire video, the user can decide to first view the summary (a sequence of keyframes) of the video, which is created dynamically by our content services network at the edge of network. This video summary causes very little bandwidth to transmit and offers a table of content to index into individual video segment for nonlinear streaming [18]. In addition, an adaptive frame dropping [20] could be also used to improve the user experiences. This service is available for both stored and live video.

Figure 11 shows an example of our adaptive video delivery service to a variety of mobile devices. As can be seen, a live MPEG2 video stream of a meeting room recorded by a camera is transmitted to a smart phone (Microsoft emulation), a handheld computer (HP Jornada 720), and a Pocket PC (Compaq iPAQ 3670). The video adaptation performs a modality transform to convert the live video stream into a sequence of keyframes which summarize the video. As these devices are simply not powerful enough to decode a live MPEG2 video, the video summary provides the user an efficient way to access the most important information in the video. This service is valuable to a lot of scenarios in the wireless and mobile space.

Our keyframe extraction algorithm is based on a triangle model of perceived motion energy (PME) developed in our lab [19]. In our

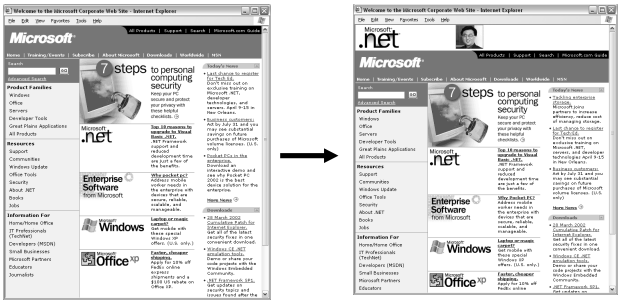


Figure 12. Personalized content is inserted at the top of web page according to the user specified context. In this example, the link to the lab's internal homepage is provided to the clients from Microsoft Research Asia.

prototype system, the user can also set his preferred density of keyframes to achieve the best user experience.

4.3.3 Personalization Service

A personalized content insertion is also enabled in our system. When subscribing to this service, the user can specify his interest on receiving what types of content (e.g., news, links, or images) inserted on the top of his requested web page.

Figure 12 shows the personalized version of Microsoft homepage. The service-enabled proxy detects that the current client comes from Microsoft Research Asia, and therefore, a photo of the lab director and the link to the lab's internal homepage is automatically added to the header.

We are currently exploring the best mechanism for content providers to leverage such personalization service. We would like them to be able to subscribe to this service so that personalized advertisement can be added to their content intelligently. Our ultimate goal is to hook up with the myProfile service of .NET My Services so that content providers could make use of the user's profile information to insert a more relevant AD.

4.3.4 Language translation

We integrate Altavista's BabelFish translation service [2] into our system. It supports language translation among English, Chinese, French, German, Italian, Japanese, Korean, Portuguese and Spanish. Since it does not provide a standard interface for web service, we only use its CGI interface for a demo purpose.

The user can specify his preferred language to translate the web page. The requested web page is sent to Altavista's BabelFish for translation by our service-enabled proxy. After the translation is completed, it is then forwarded to the user. This whole procedure can be made transparent to end users. Figure 13 shows the result of English-to-Chinese translation of the Microsoft homepage. Content providers can also subscribe to this service to make their content readable to more people. The language translation service demonstrates the usefulness of a common service delivery platform to compose and incorporate services provided by third-party service providers.

5. RELATED WORKS

The service-enabled proxy is related to the IETF working groups on Open Pluggable Edge Services (OPES) [27], which is addressing the problem of extending the functionality of a caching

(a)

(b)

Figure 13. Language translation service provided by our system. The above example shows the result of English-to-Chinese translation for Microsoft homepage. (a) Original web page. (b) The result of translation.

p

roxy for providing additional services that mediate, modify, and monitor object requests and responses. In this working group, other relevant components such as Intermediary Rule Markup Language (IRML) and the remote call-out protocols such as the use of SOAP or Internet Content Adaptation Protocol (ICAP) [17] are being discussed. Edge Side Include (ESI) [11] is a mechanism allowing an embedded script in the HTML file to instruct a proxy server at the edge of network to assemble content accordingly.

Adaptive content delivery has also been studied quite extensively in the past few years. Example works include Spyglass [32], ProxiNet [28], Intel QuickWeb [16], IBM Transcoding proxy [30], WBI [5], UC-Berkeley TranSend [12], Digestor [6], Mobiware [4], and Smart Client [36]. More recently, iMobile [29] investigated the issues of building mobile services using a proxy. The proxy maintains user and device profiles, accesses and processes internet resources on behalf of the user by performing content transformations according to the device and user profiles. Compared to content services network, these works did not address the service model and necessary mechanisms to make it a web middleware service available to content providers and end users.

Other related works include BARWAN [7] which studied the cluster-based computing for Internet services like content adaptation and caching. Active Cache [8] and Gemini [26] studied the possibility of delivering some code from server to the proxy in order to have a better control over the cached content. Active Names [35] provides a flexible resource location and transport solution taking advantage of DNS-like name resolution mechanism. Services can be associated with names and resources and executed over them for customized location and transport. The problem with this work is that it is not transparent to existing web applications.

Active Network [33] intends to improve the extensibility of the network by putting computations inside into routers. Active Service framework [3] restricts such computations to those not affecting packet routing semantics and provide a scalable cluster platform for hosting services/computations. Services in both approaches usually perform packet-level operations and are not suitable to handle application-level objects such as web pages.

CANS [13], ICEBERG [22] and Ninja [14] explore service composition and achieve service portability by means of path construction and adaptation from existing or discovered intermediary services. Services such as simple content format and transport protocol transformations were considered.

6. SUMMARY AND FUTURE WORK

In this paper we presented our work on delivering content-oriented web services to Internet media. We described the framework of content services network which consists of a new service delivery layer and the service-enabled proxy in the CDN layer. These two layers interact with each other collaboratively to provide content-oriented processing into the current content delivery path. The services are enabled based on a subscription model which allows content providers and content consumers to easily leverage this resource at the edge of Internet to deliver or access information more effectively.

To demonstrate the idea and evaluate our system design principles, a system called Media Companion has been developed and deployed in our corporate network. We provided five different services for people to subscribe to, including the transcoding of large web pages to facilitate web browsing on small devices, adaptive delivery of stored and live video streams based on video summarization, personal content insertion, and language translation service. These services have shown to be very useful to our users. Our next step is to further expand the types of services by incorporating more media technologies into our system. Our goal is to eventually make advanced multimedia processing, analysis, and interactive delivery technologies part of the Internet infrastructure services that can be easily accessed and used like commodity by anyone from any device.

Many research issues for realizing such a goal still remain uninvestigated. For example, we need to further work on the problem of dynamic service distribution and management based on a set of metric (the estimated demand of services, the history of servers' load, and demography). How to enable service composition, federation, or service-peering is another interesting and important research direction. Furthermore, the issue of data integrity also needs to be addressed as the content is going to be modified by a third-party service provider. There is a need to develop a proper security mechanism and trust model to ensure the content is modified in a desirable way. We will continue to work on these issues in the future.

7. ACKNOWLEDGMENTS

We are thankful to Zheng Zhang for many valuable suggestions in shaping up this work.

8. REFERENCES

[1] Akamai. http://www.akamai.com/

[2] Altavista – World/Translate. http://world.altavista.com

[3] E. Amir, S. McCanne, R. Katz. An Active Service Framework and its Application to Real-time Multimedia Transcoding. Proceedings of ACM SIGCOMM '98, Vancouver, British Columbia, Sep 1998

[4] 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

[5] R. Barrett and P.P. Maglio. Intermediaries: An approach to manipulating information streams. IBM Systems Journal, 38:629-641, 1999

[6] 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

[7] E.A. Brewer, R.H. Katz et al. A Network Architecture for Heterogeneous Mobile Computing. IEEE Personal Communications, October 1998

[8] P. Cao, J. Zhang and K. Beach. Active Cache: Caching Dynamic Contents on the Web. Proceedings of IFIP International Conference on Distributed Systems Platforms and Open Distributed Processing (Middleware '98), pp. 373-388, 1998

[9] E. Christensen, F. Curbera, G. Meredith and S. Weerawarana. Web Services Description Language (WSDL) 1.1. W3C Note, March 2001. http://www.w3.org/TR/wsdl

[10] Digital Island. http://www.digitalisland.com/

[11] Edge Side Includes (ESI) Web Page. http://www.esi.org/

[12] A. Fox, S.D. Gribble, Y. Chawathe and E.A. Brewer. Adapting to Network and Client Variation Using Infrastructural Proxies: Lessons and Perspectives. IEEE Personal Communications, 5(4):10-19, Aug. 1998

[13] X. Fu, W. Shi, A. Akkerman and V. Karamcheti. CANS: Composable, Adaptive Network Services Infrastructure. USENIX Symposium on Internet Technologies and Systems (USITS), March 2001

[14] S.D. Gribble, M. Welsh, R. von Behren, E.A. Brewer, D. Culler, N. Borisov, S. Czerwinski, R. Gummadi, J. Hill, A. Joseph, R.H. Katz, Z.M. Mao, S. Ross and B. Zhao, The Ninja Architecture for Robust Internet-Scale Systems and Services. Computer Networks: Special Issue on Pervasive Computing, Vol 35, No. 4, pp473-497, Mar. 2001

[15] M. Gudgin, M. Hadley, J.-J. Moreau and H.F. Nielsen. SOAP Version 1.2 Part 1: Messaging Framework. W3C Working Draft, October 2001. http://www.w3.org/TR/soap12-part1/

[16] Intel QuickWeb. http://www.intel.com/quickweb

[17] Internet Content Adaptation Protocol (ICAP) Web Page. http://www.i-cap.org/

[18] S.-J. Lee, W.-Y. Ma and B. Shen. An Interactive Video Delivery and Caching System Using Video Summarization. Computer Communications, vol. 25, no. 4, March 2002, pp. 424-435.

[19] T.M. Liu, H.J. Zhang and F.H. Qi. A Novel Video Key Frame Extraction Algorithm. Accepted by ISCAS 2002, Arizona, 2002

[20] T.M. Liu, H.J. Zhang and F.H. Qi. Modeling User Satisfaction to Frame Dropping and Its Application in Adaptive Video Delivery. Microsoft Research Asia Internal Report, 2002

[21] W.-Y. Ma, B. Shen and J. Brassil. Content Services Network: The Architecture and Protocols. Proceedings of the Sixth International Workshop on Web Caching and Content Distribution, June 2001

[22] Z.M. Mao and R.H. Katz. Achieving Service Portability using Self-adaptive Data Paths. IEEE Communications Magazine special Issue on Service Portability and Virtual Home Environment, January 2002, pp. 108-114

[23] Microsoft .NET My Services. http://www.microsoft.com/myservices/

[24] Microsoft Internet Security and Acceleration Server, http://www.microsoft.com/isaserver/

[25] Mirror Image. http://www.mirror-image.com/

[26] A. Myers, J. Chuang, U. Hengartner, Y. Xie, W. Zhuang and H. Zhang. A Secure, Publisher-Centric Web Caching Infrastructure. Proceedings of Infocom '01, 2001

[27] Open Pluggable Edge Services (OPES) Web Page. http://www.ietf-opes.org/

[28] ProxiNet. http://www.proxinet.com

[29] C.H. Rao, Y.R. Chen, D. Chang and M. Chen. iMobile: A Proxy-Based Platform for Mobile Services. Proceedings of the First ACM Workshop on Wireless Mobile Internet (WMI2001), Rome, Jul. 2001

[30] J. Smith, R. Mohan and C. Li. Scalable multimedia delivery for pervasive computing. ACM Multimedia, 1999

[31] SOAP Toolkit 2.0. http://msdn.microsoft.com/library

[32] Spyglass-Prism. http://www.spyglass.com

[33] D.L. Tennenhouse, J.M. Smith, W.D. Sincoskie, D.J. Wetherall and G.J. Minden. A Survey of Active Network Research. IEEE Communications Magazine, Vol. 35, No. 1, pp80-86. Jan. 1997

[34] Universal Description, Discovery, and Integration (UDDI) Web Page. http://www.uddi.org/

[35] A. Vahdat, M. Dahlin, T. Anderson and A. Aggarwal. Active Names: Flexible Location and Transport of Wide-Area Resources. Proceedings of the Second USENIX Symposium on Internet Technologies and Systems, Oct. 1999

[36] C. Yoshikawa et al. Using Smart Clients to Build Scalable Services. Proc. Winter 1997 USENIX Tech. Conf., January 1999