WOD – Proxy-Based Web Object Delivery Service

Kai-Hsiang Yang and Jan-Ming Ho

Institute of Information Science, Academia Sinica
{khyang, hoho}@iis.sinica.edu.tw

Abstract. With the tremendous growth of World Wide Web (WWW), the door has been opened to a multitude of services and information for even the most casual of users. Today, many wireless and mobile devices are being produced to provide access to this information, and the capabilities of these devices can vary depending on characteristics such as physical memory, storage space, and network speed. In the future, it is expected to see a rich variety of devices that can browse the WWW, and any given user is likely to own more than one type. When a user browses the WWW by small handy devices, such as PDAs or mobile phones with low network bandwidth, the storage space limitation and long download time make a user unable to download large-size web objects such as software zip files. One possible solution is for the user to memorize the URL of the desired web object, and download it when he reaches his home or office computer, but this is extremely inconvenient, and in most cases highly impractical. In this paper, we propose that using a proxy-based web object delivery system is a much more convenient and efficient solution. The proposed system is actually an HTTP proxy server that automatically checks all the requested web objects according to user-defined rules. If one or more rules are found to match, and the web object needs to be delivered to the user's account, the proposed system does some translations for the web object depending on its Content Type, and then delivers it via the Simple Mail Transfer Protocol (SMTP) or File Transfer Protocol (FTP). Users need merely set up the rules, and the web objects can proceed to be sent to their email or ftp site. There it can be downloaded at the user's leisure in high speed network environments such as at home or at one's office. In addition, a scheduling mechanism has been designed in order to enhance performance and improve the quality of service (QoS) for the users. We have tested the proposed system on the Windows platform, and have also evaluated it by a Pocket PC emulator.

1 Introduction

The era of PC-dominated applications is coming to an end. Today, we see widespread use of mobile devices that have sufficient computing and networking capabilities to browse the Web. The network-enabled devices come in a variety of types, including cell phones, Personal Digital Assists (PDAs), Pocket PCs, handheld PCs, car navigation systems, and notebook PCs. An average person may already own several of such devices, and it is expected that in the near future, the number of such devices will far exceed the number of desktop PCs. These devices have different capabilities due to varying processors, physical memory, network protocols, screen sizes, input methods, software libraries, and more. And with this variety of devices, certain inconveniences arise for users who want to browse the web. This paper focuses on a problem users often encounter while on their wireless and mobile devices; the problem is described in the following section.

The current network-enabled devices are becoming more and more diverse. However wireless and mobile devices with small memory and lower network bandwidth are unsuitable for complicated computations and downloading large files. For example, a user may want to surf the net using a mobile phone while taking the mass transit system. Due to low connection speed and the fact that the user is mobile, interruptions may occur, causing the user to only be able to access simple web-based information. This would prevent him from retrieving more abundant information such as business achievement reports or zipped software files. Another challenge that users might face is the low speed of the public wireless connection or of the dial-up services through PHS- or GPRS-type mobile phones. Because these services charge based on the amount of information downloaded, users are less likely to want to retrieve information or files of larger size. These problems may result from the following situations: (1) Small storage - If the devices such as Palms of Pocket PCs can only contain a limited amount of information, large-sized files are impossible to download. (2) Differing web page protocols and file format support - Some devices, such as mobile phones, contain browsers different from general web page protocols and file format support. There are even some special embedded systems that only accept pure text files, and no other complicated file formats. (3) Low network bandwidth and user mobility - When users are moving or connection speed is low, it becomes extremely difficult to directly download files of large size. (4) Browsing a web site with slow network speed - When users surf web sites that have slow network speed, download time will be very long and the connection will likely time out. This kind of problem is called the ”web object downloading problem”.

Because of this problem, users need to find a better solution than directly downloading web objects in the current conditions. The simplest solution is to have users memorize or write down the URL of the web object, and have them download it when they arrive someplace where the network speed is fast and stable. However this solution has three obvious drawbacks: The first is that it is cumbersome to have users do such a thing, and there is always the possibility of misspelling or forgetting the URL, especially when the URL is long and contains many unmeaning symbols. The second drawback is that, the user cannot initiate the download until he reaches the place where network speed is faster. As a result, time is lost waiting for files of large size to finish downloading. Third, some web objects cannot be directly downloaded by a URL because of the security policy of web sites. Users have to repeat those serial of browsing actions to be able to download the objects.

In this paper, we propose a proxy-based web object delivery (WOD) system as the solution for this problem. This system is a personal proxy server that can be deployed for personal use or as a general network proxy server. When users use different wireless and mobile devices, by setting the WOD as the default proxy server, the WOD system automatically checks all the requested web objects against the user's rules. If there is one rule matched and the web object needs to be delivered to the user's account, the proposed system does some translations for the web object depending on its Content Type, and then delivers it via the Simple Mail Transfer Protocol (SMTP) or File Transfer Protocol (FTP). Users can browse the web object later by checking their email or connecting to their ftp site when they are in a high speed network environment.

In the proposed system, we have designed a rule model suitable for most situations, which focuses on the Content Type and Content Length of web objects. These rules can be set for one or more delivery methods, such as email, ftp, or for several methods simultaneously. Users can change the delivery rules any time, and the WOD system immediately follows the rules to work. Also, taking into consideration system performance, to improve quality of service (QoS) we have developed a delivery scheduling component to schedule all the delivery tasks according to their priorities. Delivery scheduling can prevent deadlock and allow the high priority tasks to be delivered more quickly. The proposed system is implemented under the Windows platform, and based on the proxy module of the Apache web server. We used a MySQL database to store each user's rules, web objects, and delivery status, etc. For the user profile management aspect of the system, we used PHP scripting language to implement a rule management page; this is for rule setup and for checking delivery task statuses.

This paper is organized as follows: Section 2 discusses related works. Section 3 presents an overview of the proposed web object delivery service. Section 4 describes the delivery rule model we used. In section 5 we present the delivery scheduling model. Section 6 describes the implementation details and related technologies. And section 7 is the conclusion.

2 Related Work

There are some commercial products [1, 2, 3] and research works [4, 5] for the problem of heterogeneous client devices. These products focus on providing quality of service and performing content transformation in proxy of a variety of client devices through a process called "transcoding." By maintaining separate caches for different categories of clients, such as PC, PDA, Mobile, etc., it is possible to translate large size objects to small size ones at the proxy servers. Besides, some works [6, 7, 8] focus on Web page layout modification techniques to fit the mobile devices. However, although the transcoding technique solves part of the problem, such as the issue of image translation, there are a couple of problems that can arise. First, there are objects that cannot be translated by transcoding, such as zip files. Next, there are problems that occur when clients download web objects that cannot be handled by their devices. But our proposed system can handle these problems. This system filters these web objects according to user-defined rules, and delivers the objects to the user's email account or ftp space; the web objects are not downloaded directly into the client device so users can access them later from the medium of choice.

Generally, current proxy systems applying conventional page-level caching cannot function effectively for those larger and dynamic web objects. These proxies can be configured to download web objects for users even if the request connection is closed in the middle of a transfer. The proxy systems will complete the transfer to the cache if it has already transferred more than a specified percentage. Generally a number between 60% and 90% is usually what's recommended. However, due to space consideration, general proxy systems will have a higher percentage setting. Another important setting in proxy is the max cache size. Web objects with size greater than the max cache size will not cached by the proxy. Even if the proxy caches some of such web objects, they will most likely be replaced by the cache replacement mechanism before the user has a chance to download them later. Several new caching strategies [9, 10, 11, 12] have been proposed, where a fragment-level caching strategy is applied, to solve the problems. A recent work [13] focuses on integrating a Web content adaptation algorithm and a caching mechanism to serve dynamic content in a mobile computing environment. However, one disadvantage of using existing proxy systems is that the user has to download these objects from the proxy that his device connected to. A user will probably want to use the device outside and then browse those web objects later in his office with high speed network; he will not want to remote connect to the proxy his device connected to. Our proposed system can solve this problem by delivering the web objects to user's email account or ftp space close to him. After the web objects are delivered, proxy can delete them immediately without affects on the space and performance.

3 Proposed System

This section will introduce the web object delivery (WOD) system, including its logical concepts, system architecture, rule model, scheduling, and delivery modules.

3.1 Logical Concepts

Figure 1 illustrates the logical concepts of the WOD system. The circle on the right is the network environment in the company or organization. Each user has his mail account and ftp site in the email server and ftp server. The circle on the left represents the locations such as a user's home, a coffee shop, or even a train station; places where users may use different devices to surf the Internet. If the Internet Service Provider (ISP) provides WOD for users, the web object-downloading problem will be easily solved. Users can initially define which web objects have to be delivered, and where they should be delivered. After the rule setup, users can start surfing the Internet via the WOD system. When the

WOD system receives a request, it tries to get the web object and its header information. According to the Content-Type and Content-Length in the header, the proposed system checks the user's rules to decide whether or not the web object should be delivered. If the web object has to be delivered, the system first does the proper translations, and then delivers it via email or ftp. This way, when users are mobile or in a slow network speed environment, they can deliver important files, software zip file, specification PDF files, or even mp3 files to their accounts the WOD system provided by the ISP. This solution will prevent users from having to write down URLs, and will also save users the download time by delivering to users' accounts.

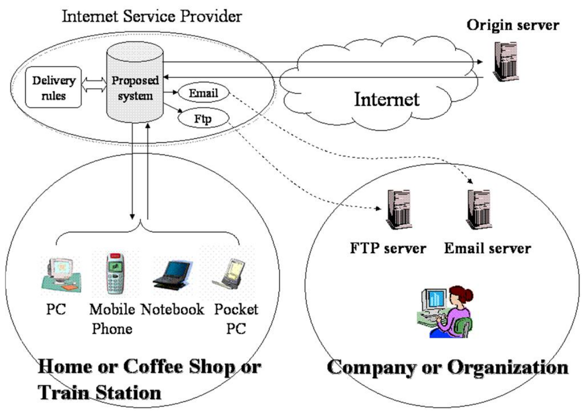


Fig. 1. Logical Concepts

3.2 System Architecture

This section details the system architecture of the WOD. Figure 2 illustrates all the elements in the WOD system. When receiving a request, the system tries to get the web object and its header information from the cache or origin server. After saving the web object into the cache, the system checks the user's rules to decide whether the web object has to be delivered. The six important elements in this system are described in the following sections.

Proxy authentication. To distinguish users, the authentication mechanism is necessary for the WOD system. The HTTP 1.0/1.1 protocols provide a simple challenge-response authentication mechanism which is used by the proxy to challenge a client request by requiring that the client provide authentication information. It uses an extensible, case-insensitive token to identify the authentication scheme. A 407 (Proxy Authentication Required) response message is used by a proxy to challenge the authorization of a client. The response includes a "Proxy-Authenticate" header field containing the information for the proxy authentication. This proxy authentication mechanism is supported by almost every existent browser. Hence, the WOD system applies this standard mechanism to authenticate users.

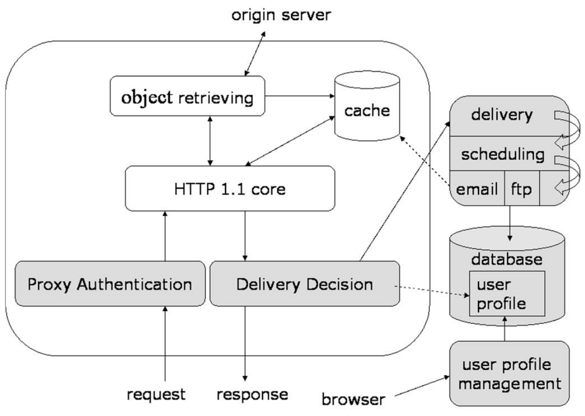


Fig. 2. System Overview

Delivery decision element. The Delivery Decision (DD) element is used to decide whether the requested web object has to be delivered, and guarantees the robustness of the proposed system by determining if each step has been successfully completed or not. This element only takes care of the decision-making, but does not play a part in the action of making a delivery. To increase download speed help make the network connections reliable, the general proxy server returns all receiving packets to users immediately after it receives parts of requested web objects. This method will help to prevent connections timeouts when users attempt to download large-size files. Therefore, the DD element must make decision in a short amount of time. If the requested web objects need to be delivered, the DD element returns the delivery messages to users. Otherwise the DD element has to work as the conventional proxy server. If the web objects need to be delivered, the DD element first stores necessary information into the database, then immediately closes the client connections. But the connections to origin servers are still preserved so it can continue to receive the web objects. After the web objects have been completely delivered, the DD element updates the status of web objects, including the cache file paths and the delivery accounts.

Delivery element. The delivery element ensures that each delivery file is complete and ready to be sent to the 'Scheduling' element. It is revoked every several minutes, checks whether all web objects are complete, and then checks that their Content Types and Content Lengths are also correct. Also, for the web objects that expire at the delivery time, the element will mark those filenames with the notation "[expired]". Most importantly, only the tasks with complete files will be processed. This way, a preceding large-file download task will not affect later small-file download tasks. Before sending tasks to the Scheduling element, the delivery element has to compute the priority of each task. We define 10 different user levels from 1 to 10, and each user level has 10 units of priority. Also, users can set each delivery rule different priority from -5 to +5. For example, one user is the user level 3, and sets the first delivery rule with the priority '+4', and the second delivery rule to the priority '-2'. According to the above settings, the delivery tasks that satisfy the first delivery rule have the priority 34 (3 * 10 + 4 = 34), and those that satisfy the second delivery rule have the priority 28 (3 * 10 - 2 = 28).

Scheduling element. The scheduling element focuses on delivering tasks according to their priorities from highest to lowest. The element also updates the priority of the tasks that have been delivered failed, in order to prevent the deadlock situation. All the details are described in section 5.

Database. All the information about the delivery tasks is stored in the database. By using the database, the proposed system can easily manage the data and provides users with the status of their delivery tasks.

User Profile Management. This management element provides users an interface to customize their preferences. Through the element, users can set their rules, delivery accounts, methods, and also view the historical logs and status of delivery tasks. Aside from the convenience, users can setup different rule groups for different device properties, in order to quickly open or close some specific rules. Each rule group also can be configured to automatically open according to the user's IP.

4 Rule Model

4.1 Field Selection

In order to define one rule model which is suitable for all situations, the rule model should contain the information in the header of web object. However there are many fields in the HTTP header, how to choose the proper fields is very important. We choose the fields by the following policies: 1. The field should be in most header of web object. 2. The field is suitable for most situations. 3. Fewer fields are better. According to these policies, we choose the Content-Type and Content-Length for our rule model. Content-Type is a critical attribute of web objects for many applications. Some types of documents are probably not supported by the client device, and users can set up rules to deliver web objects with those specific Content-Types. Content-Length is another important attribute. At high network bandwidth environment, hundreds of mega bytes are allowed to download. However when client devices connect to Internet by the dial-up services through PHS- or GPRS-type mobile phones, only web objects with few Kbytes are allowed to download. Moreover, to download the large-size web objects, the stable connection is necessary otherwise the download process may be going to fail. However in the wireless network, the connection usually is not stable.

4.2 Positive List of Rules

According to the above consideration, a positive list of rules is used in the proposed system for the following two advantages.

1. Low complexity
   The system cannot spend much time on each task for deciding the delivery method. There may be some advanced methods which could make better decision, but they are usually complex. A positive list of rules can make the decision quickly even with a large number of requests.
2. Clear definition
   Another benefit of the positive list of rules is the definite result. It is essential and brings better performance. Basically, it is not expected if there are too many rules to be triggered. Therefore, only the first matched rule is triggered once. There are two advantages of this design. (1) Simplify the rules: each rule can be made shorter, and several rules are grouped to achieve one goal. The speed of checking rules doesn't decrease because not all of them are checked. Actually the checking time is less than the time for checking a long rule. (2) Avoid contradiction: if some rules are matched with the contradiction, it certainly makes troubles. Our design can prevent the situation by only trigger one rule. Figure 3 shows the rule checking flow. Each rule contains four fields, including an owner ID, a Content-Type, a Content-Length, and an Action, where the action has two values, 'D' refers to deliver the web objects and 'R' refers to return the web objects to clients. Assume there is one web object W, and rules R_j, where j is from 1 to n. If (W's Content-Type matches R_j's) and (W's Content-Length is greater than R_j's) then the rule will be applied.

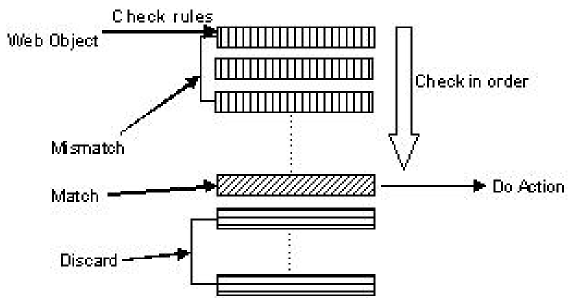


Fig. 3. Rule Checking Flow

4.3 Example

Assume that client connects to the Internet by the dial-up services, and the slow speed just accepts browsing the normal web page. The size of web page usually is smaller than 1MB; therefore the following rules are set for this situation.

{Owner,Type,Length,Action}={"Jacky","*",1000000,"D"}

Any web objects with Content-Length larger than 1MB will be delivered. This also shortens the download time and saves the network bandwidth.

5 Scheduling

In this section, we will discuss the scheduling element. Generally speaking, the delivery tasks are produced in a fast speed. To prevent from affecting the system performance and provide the quality of service (QoS), it is necessary to develop one scheduling mechanism to schedule these delivery tasks.

5.1 Purpose

There are several purposes for implementing the scheduling element. (1) Without reducing the system performance: the system performance must be considered foremost during our design. The delivery process may be very fast or very slow, so the delivery time is always unexpected and the system can't deliver the tasks on line. (2) Priority support: the proposed system wants to provide the quality of service for different users. Therefore each task has its own priority and is scheduled by the scheduling element.

5.2 Scheduling Mechanism

The concept of scheduling mechanism is shown in Figure 4. The scheduling mechanism contains a dispatcher, multiple priority queues, a selector, and a module interface.

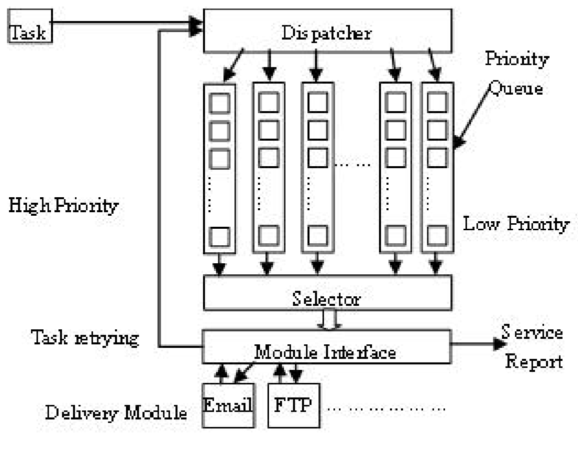


Fig. 4. Scheduling Model Diagram

When a task comes in, it is dispatched by the dispatcher element to the corresponding priority queue according to its priority. The left priority queues have higher priority than the right ones. The selector element chooses the tasks from the priority queue and sends them to the specific delivery modules. All delivery modules are external and are managed by one module interface. The external modules communicate with the interface to get the delivery web objects and return the status of delivery tasks. Some special issues are described as follows.

5.2.1 Batch Processing

In order to make the system resources perform more efficiently and do not reduce the proxy performance, the scheduling element is revoked every several minutes, and all the tasks will be batch-processed.

5.2.2 First-In-First-Out

In each priority queue, the delivery order is FIFO to ensure that all tasks in the same priority queue are processed by order. The FIFO design and again mechanism can prevent the deadlock situation.

5.2.3 Aging Mechanism

The aging mechanism is used to deal with some exceptions. When the task gets in trouble during delivering, the scheduling element has to recompute its priority. If one task is delivered with errors, we believe that it cannot be delivered in a short time to prevent encountering the same errors. Therefore, the priority of the task will be decreased as shown in Figure 5. If the task is delivered with error many times, the system will drop it and reports to users. If one task is waiting for a long time, the priority of it will be increased. The aging mechanism is to guarantee the system robustness by checking each task with error or not.

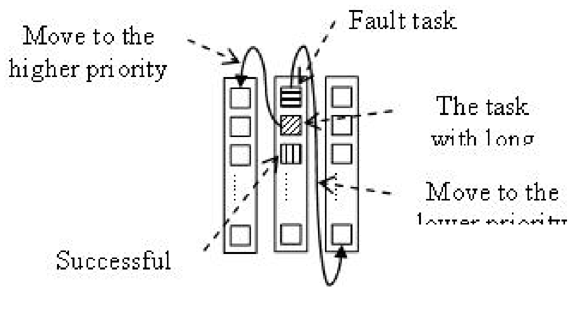


Fig. 5. Aging Mechanism

5.2.4 Module Interface

The module interface aims to integrate different delivery modules, control these modules' behavior and immediately report the current status of delivery tasks.

6 Implementation and Experimental Results

In this section, we describe the implementation details and present the experimental results of testing on the Pocket PC emulator in the Microsoft embedded visual tools. We start in section 6.1 by describing the system environment that the proposed system is implemented. In section 6.2 we discuss the implementation issues of each component. Finally, we present the experimental results in section 6.3.

6.1 System Environment

Our implementation is on the Windows platform, and is based on the proxy module in the Apache web server. For the considerations about system scalability and security, the MySQL database locates on another computer under the RedHat Linux platform. This design can improve the system reliability, and is easy to extend to a cluster architecture. We used the Visual C++ to develop the whole proxy system, and used the PHP language to develop the user profile management system.

6.2 Implementation Issues

The implementation issues of each component are described as follows:

1. **Proxy Authentication.** The Proxy Authentication component is designed to follow the HTTP 1.0/1.1 standard. When one request comes in, the WOD system checks its username and password pairs with the valid user data in the MySQL database. If they do not match, an error message is sent to client. After the authentication, the username is saved in the connection structures for the future use.

2. **Delivery Decision element.** This element is called by the proxy core, and is responsible for checking in a very short period of time whether the request web objects should be returned to users. A proxy probably has many threads corresponding to requests at the same time, and each thread must decide every decision as soon as possible. If the web objects should be returned to users, the DD element returns small packets immediately when it has been received, without waiting for the file to complete being sent.

3. **Delivery element.** This element is also called by the proxy core. It has to confirm the integrity of web objects. After the web objects are complete, it assigns a priority to these tasks according to the user's settings, and then sends the tasks to scheduling element.

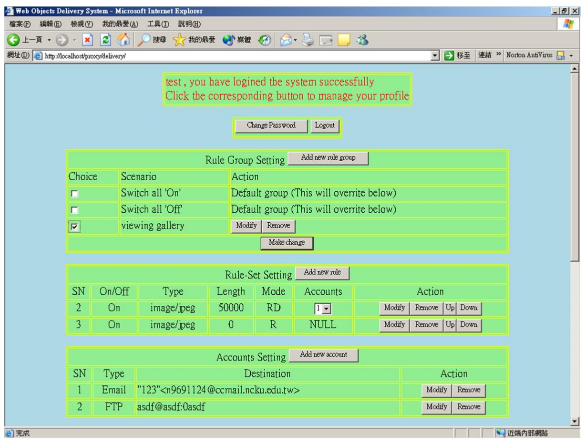
4. **Scheduling element.** In our system, the priority queues are divided into 10 different queues. The first queue, Q1, contains the tasks with priorities between 1 and 10, and Q2 contains the task with priorities between 11 and 20, etc. We also assign each user a priority from 10 to 100.

5. **Database.** We choose the MySQL database as our storage element. When a large number of requests come, the proxy server creates many threads for handling these requests at the same time. Each thread has to connect to the MySQL database for checking the rules and storing the web objects for delivery. In order to avoid opening large number of connections simultaneously, we develop all threads using one connection, and create a critical section between the beginning of each query and the ending of storing the results.

6. **User profile manager.** We use the PHP to develop the user profile management, and for the convenience, users can input the system IP address like "http://proxy-ip/" to setup their rules.

6.3 Experimental Results

In this section we present the experimental results by showing the user management and the physical situations tested by the Pocket PC emulator software. After user authentication, valid users can see the following management page like Figure 6. Users can configure their own rules and accounts, switch each rule on and off, and group some rules as one rule group. All these operations are very easy for users.



Web Objects Delivery System - Microsoft Internet Explorer
檔案(F) 編輯(E) 檢視(V) 我的最愛(A) 工具(T) 說明(H)
上一頁 搜尋 我的最愛 媒體
網址(D) http://localhost/proxy/delivery/
移至 連結 Norton AntiVirus
完成 近端內部網路

test , you have logined the system successfully
Click the corresponding button to manage your profile

Change Password | Logout

Rule Group Setting — Add new rule group

Choice	Scenario	Action
☐	Switch all 'On'	Default group (This will overrite below)
☐	Switch all 'Off'	Default group (This will overrite below)
☑	viewing gallery	Modify Remove

Make change

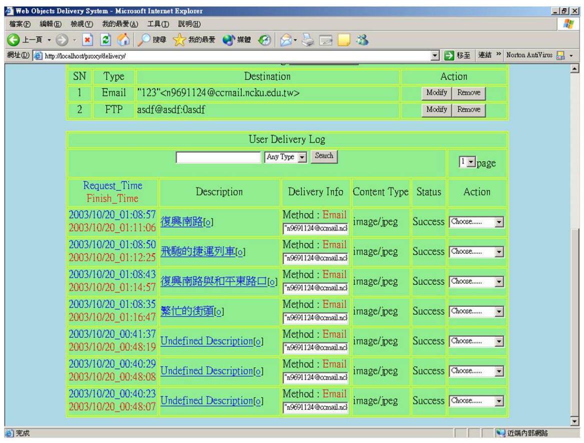
Rule-Set Setting — Add new rule

SN	On/Off	Type	Length	Mode	Accounts	Action
2	On	image/jpeg	50000	RD	1	Modify Remove Up Down
3	On	image/jpeg	0	R	NULL	Modify Remove Up Down

Accounts Setting — Add new account

SN	Type	Destination	Action
1	Email	"123"<n9691124@ccmail.ncku.edu.tw>	Modify Remove
2	FTP	asdf@asdf:0asdf	Modify Remove

Fig. 6. User Profile Management System



Web Objects Delivery System - Microsoft Internet Explorer
檔案(F) 編輯(E) 檢視(V) 我的最愛(A) 工具(T) 說明(H)
上一頁 搜尋 我的最愛 媒體
網址(D) http://localhost/proxy/delivery/
移至 連結 Norton AntiVirus
完成 近端內部網路

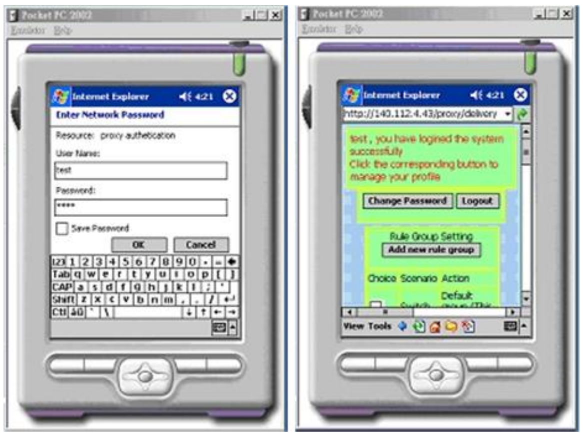
SN	Type	Destination	Action
1	Email	"123"<n9691124@ccmail.ncku.edu.tw>	Modify Remove
2	FTP	asdf@asdf:0asdf	Modify Remove

User Delivery Log

Any Type | Search | 1 page

Request_Time Finish_Time	Description	Delivery Info	Content Type	Status	Action
2003/10/20_01:08:57 2003/10/20_01:11:06	復興南路[o]	Method : Email "n9691124@ccmail.ncl	image/jpeg	Success	Choose......
2003/10/20_01:08:50 2003/10/20_01:12:25	飛馳的捷運列車[o]	Method : Email "n9691124@ccmail.ncl	image/jpeg	Success	Choose......
2003/10/20_01:08:43 2003/10/20_01:14:57	復興南路與和平東路口[o]	Method : Email "n9691124@ccmail.ncl	image/jpeg	Success	Choose......
2003/10/20_01:08:35 2003/10/20_01:16:47	繁忙的街頭[o]	Method : Email "n9691124@ccmail.ncl	image/jpeg	Success	Choose......
2003/10/20_00:41:37 2003/10/20_00:48:19	Undefined Description[o]	Method : Email "n9691124@ccmail.ncl	image/jpeg	Success	Choose......
2003/10/20_00:40:29 2003/10/20_00:48:08	Undefined Description[o]	Method : Email "n9691124@ccmail.ncl	image/jpeg	Success	Choose......
2003/10/20_00:40:23 2003/10/20_00:48:07	Undefined Description[o]	Method : Email "n9691124@ccmail.ncl	image/jpeg	Success	Choose......

Fig. 7. Delivery histories



Enter Network Password

Resource: proxy authetication

User Name: test

Password: ****

Save Password

http://140.112.4.43/proxy/delivery

test , you have logined the system successfully
Click the corresponding button to manage your profile

Rule Group Setting

Choice Scenario Action

Fig. 8. Proxy authentication mechanism and user profile management pages

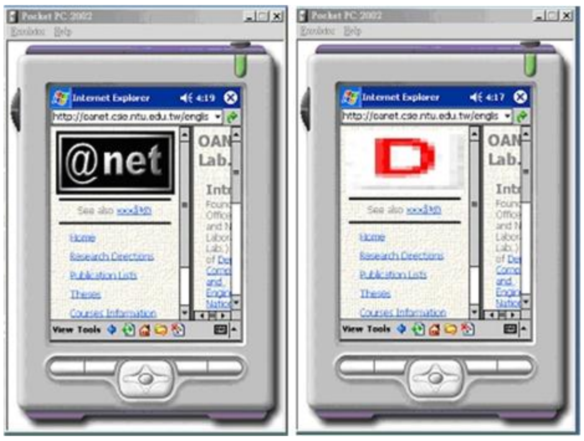


Fig. 9. Situations before and after using the WOD system

Besides, users can see the delivery histories, access time, delivery time, and other related information in this page. Figure 7 shows the delivery histories. For each delivery record, users even can write the comments for it to explain what it is. One simple search engine is implemented to help users finding the records by searching the comments.

We then used a Pocket PC emulator to test the WOD system. After setting the wireless connections and assigning the WOD system as default proxy server, a proxy authentication page is shown up (the left picture in Figure 8). When users pass the authentication, they can browse the Web as usual. Figure 8 also shows the user profile management page in Pocket PC.

After users set up a rule to deliver all images large than 50 Kbytes, the WOD system immediately works. Figure 9 shows a page before and after setting the rule. In the right picture of figure 9, a large picture is replaced by a small picture "D" to notify users that the picture is delivered by the system.

7 Conclusion

In this paper, we have presented a proxy-based system that automatically delivers web objects for the assortment of devices a client may own. It is the middleware between the client and original server, and the client does not need to install any software. We have also designed a simple rule model which is sufficient for all situations. By following the users' rules, the proposed system filters and delivers web objects via email or ftp protocols, and users can download them later at other working environments. The system also provides a user management system for users to set up and change their delivery rules, and it keeps a delivery log for users to manage and search their rules. Lastly, in consideration of the performance of the proposed system, a delivery scheduling scheme has been designed to optimize the delivery process. The delivery scheduling can prevent the deadlock and it improves on quality of service.

As well as developing a Web object delivery system, which we are now doing, there are several promising directions for future research. The file distribution has been an intensively studied research topic in the past few years. One of those established technologies is the Content Distribution Network (CDN), where a number of servers are deployed at the edge of the Internet, and clients request file download service from their closest servers. More recently, peer-to-peer (P2P) based file distribution techniques have quickly gained popularity, we plan to study how to make proxy systems into a structured or a unstructured P2P network, and combine our previous techniques on structured [14] and unstructured [15] P2P networks to enhance the system performance and scalability.

References

1. Maheshwari, A.S., Ramamritham, K., Shenoy, P.: TranSquid: Transcoding and caching proxy for heterogeneous ecommerce environments. In: RIDE 2002. Proc. of 12th IEEE Workshop on Research Issues in Data Engineering (February 2002)
2. IBM Web Intermediaries (WBI), WebSphere Transcoding Publisher, `http://www.almaden.ibm.com/cs/wbi/`
3. Bharadvaj, H., Joshi, A., Auephanwiriyakul, S.: An active transcoding proxy to support mobile web access. In: Proceedings of IEEE Symposium on Reliable Distributed Systems (1998)
4. Lara, E.D., Wallach, D.S., Zwaenepoel, W.: Puppeteer: Component-Based Adaptation for Mobile Computing. In: Proceedings of Third Usenix Symposium of Internet Technologies and Systems (March 2001)
5. Cardellini, V., Yu, P.S., Huang, Y.W.: Collaborative Proxy System for Distributed Web Content Transcoding. In: Proceedings of ACM CIKM, pp. 520–527 (2000)
6. Stuary, G., Rag, T., Sreedhar, K.: ATTENUATOR: Towards Preserving Originally Appearance of Large Documents when Rendered on Small Screen. In: Proceedings of International Conferences of Multimedia Expo 2003 (July 2003)

7. Hori, M., Kondoh, G., Ono, K., Hirose, S., Singhal, S.: Annotation-Based Web Content Transcoding. In: Proceedings of 9th Would Wide Web Conference (May 2000)
8. Chen, Y., Ma, W.Y., Zhang, H.J.: Detecting Web Page Structure for Adaptive Viewing on Small Form Factor Devices. In: Proceedings of 12th Would Wide Web Conference (May 2003)
9. Li, W.S., Hsuing, W.P., Kalashnikov, D.V., Sion, R., Po, O., Agrawal, D., Candan, K.S.: Issues and Evaluations of Caching Solutions for Web Application Acceleration. In: Bressan, S., Chaudhri, A.B., Lee, M.L., Yu, J.X., Lacroix, Z. (eds.) CAiSE 2002 and VLDB 2002. LNCS, vol. 2590, Springer, Heidelberg (2003)
10. Yagoub, K., Florescu, D., Valduriez, P., Issarny, V.: Caching Strategies for Data-Intensive Web Sites. In: VLDB 2000. Proceedings of 26th International Conference of Very Large Data Bases (September 2000)
11. Yuan, C., Chen, Y., Zhang, Z.: Evaluation of Edge Caching/Offloading for Dynamic Content Delivery. In: Proceedings of 12th Would Wide Web Conference (May 2003)
12. Zeng, D., Wang, F.Y., Liu, M.: Efficient Web Content Delivery Using Proxy Caching Techniques. IEEE Transactions on Systems, Man, and Cybernetics 34(3) (August 2004)
13. Hua, Z., Xie, X., Liu, H., Lu, H., Ma, W.Y.: Design and Performance Studies of an Adaptive Scheme for Serving Dynamic Web Content in a Mobile Computing Environment. IEEE Transactions on Mobile computing 5(12) (December 2006)
14. Yang, K.-H., Ho, J.-M.: Proof: A Novel DHT-based Peer-to-Peer Search Engine. IEICE Transactions on Communications E90-B(4), 817–825 (2007)
15. Yang, K.-H., Wu, C.-J., Ho, J.-M.: AntSearch: An Ant Search Algorithm in Unstructured Peer-to-Peer Networks. IEICE Transactions on Communications E89-B(9), 2300–2308 (2006)