ISSN:0975-9646
Harshal N. Datir et al, / (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 2 (3) , 2011, 1260-1262

Implementation of Web Cache using Web Browser

Harshal N. Datir[1] Dr. P.R. Deshmukh[2]
Department of Information Technology.Sipna's College of Engineering and Technology Amravati.
Sant Gadge Baba Amravati University Amravati (M.S.) India

Abstract : **-- Web caches will have an important role in reducing server loads, client request latencies, and network traffic. Web-caching is a well-known technique for reducing access latencies and bandwidth consumption. When a user visits a web page, the contents of that page can be stored in the browser's cache so it doesn't need to be re-requested and re-downloaded. Efficiently using the browser cache can improve end user response times and reduce bandwidth utilization. If an item is considered cacheable, the browser will retrieve the Updated contents from cache on repeat visits .**

Keywords **– Web Cache, web cache behavior, Page replacement policies**

I. INTRODUCTION

Regardless of the speed of an enterprise WAN link, the response time for retrieving content from the Internet or web varies greatly. This is because multiple factors influence response time, including the amount of bandwidth available on the internet at any given time, the amount of network traffic at any given time, and the number of requests the target server site is trying to fulfill. Caching eliminates a great deal of this delay and unpredictability by storing frequently accessed content close to the user. First, the web browser sends out a request for a Uniform Resource Locator (URL), the address of a specific web page on a specific server on the Internet. The content being requested might be a static web page (consisting of text, links and graphics files), or it might be a dynamically created page. or web-based application. When caching is implemented, frequently accessed content is stored close to the users, eliminating this duplicated effort. A request from a user's browser is first sent to the network's caching server. If the requested content can be found in the web cache and the information is current, or fresh, the content is sent directly back to the requester, skipping an upstream journey to the target website. This is called a web cache hit . If the requested object is found (a *cache hit*) the proxy can immediately respond to the client's request. If the requested object is not found (a *cache miss*) the proxy then attempts to retrieve the object from another location, such as a peer or parent proxy cache or the origin server. If the requested object is *cacheable* (based on information provided by the origin server or determined from the URL) the proxy may decide to add a copy of the object to its cache. If the object is *uncacheable* (again determined from the URL or information from the origin server) the proxy should not store a copy in its cache.

Web caching technology can be classified into following categories as follows.

Client Side Caching

Client-side caching refers to caches that are built into most web browsers, which cache Internet objects for a single user, but from a variety of servers. Caches are found in browsers and in any of the web intermediate between the user agent and the original server. Typically, a cache is located in the browser and the proxy. If we examine the preferences dialog of any modern web browser (like Internet Explorer, Safari or Mozilla), we will probably notice a cache setting. Since most users visit the same web site often, it is beneficial for a browser to cache the most recent set of pages downloaded.

Client's Browser Caching

In client's browser caching, web objects are cached in the client's local disk. If the user accesses the same object more than once in a short time, the browser can fetch the object directly from the local disk, eliminating the repeated network latency.

The cache-ability of an item on the browser is determined by:

- The response headers returned from the origin web server. If the headers indicate that content should not be cached then it won't store the cache object.
- A validator Last-Modified header must be present in the response. If an item is considered cacheable, the browser will retrieve the item from cache on repeat visits if it is considered "fresh." Freshness is determined by:
- A valid expiration time that is still within the fresh period.

If a representation is stale or does not have a valid expiration date, the browser will ask the web server of origin to validate the content to confirm that the copy it has can be served. The web server will then return a 304 to let the browser know that the local cached copy is still good to use. If the content has changed, the web server returns a 200 response code and delivers the new version.

Browser Settings

The user can configure how they want cached content to be stored and delivered from their local cache, or whether they want the content cached at all.

a) Every visit/view to the web page

When a user returns to a page that was previously visited, the browser checks with the origin web server to determine whether the page has changed since last viewed.

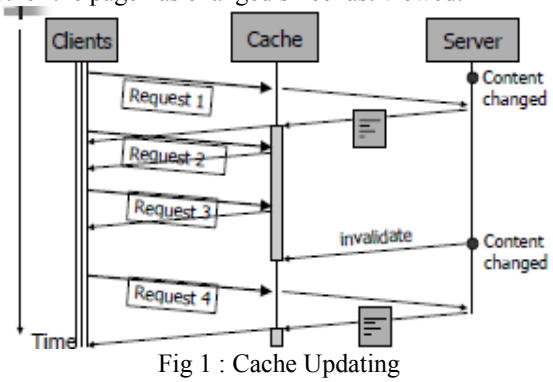

Fig 1 : Cache Updating

b) Refreshing contents Every time when start the browser/Once Per Session.

As shown in fig, If a page is revisited within the same browser session the content will be delivered from the cache. When browser is closed and then reopened, a request will be sent to check whether the content has changed. If a page is visited during the same browser session, the cached files will be used instead of downloading content from the web server of origin.

c) Automatically/ When the page is out of date

When the browser is closed and then reopened on repeat visits, it will use the lifetime settings of the cached content. If the same page is visited during a single browser session the cached files will be used. There are four values that can be used for the content variable:

- Private –May only be cached in a private cache such as a browser
- Public – May be cached in shared caches or private caches
- No-Cache – Content cannot be cached.
- No-Store – Content can be cached but not archived.

In order for content to be served from the cache, the URL has to be an exact match to the content in the cache. Some web developers will add random numbers to part of the query string to ensure that the content is not cached and is always "fresh." When these random query strings are added to the URL the browser will not recognize the content as being the same as the item already in cache and a new GET request will be issued for the element. In most instances the cache behavior of content is controlled by the Cache-Control and Expires HTTP headers. Cache-Control headers specify whether or not the content can be cached and for how long. The values can include:

- ***no-cache*** – Do not cache this content
- ***private*** – Can be cached by browsers, but not shared/public caches
- ***max-age*** – Set in seconds; specifies the maximum amount of time content is considered fresh The inclusion of just an Expires header with no Cache-Control header indicates that the content can be cached by both browsers and public/shared caches and is considered stale after the specified date and time . The Expires tag should be used in conjunction with the Cache-Control tags to specify how long content can be stored.

d) Redirecting Users to Another Page.

Refresh elements can be used to tell the browser to either redirect the user to another page or to refresh the page after a certain amount of time. The refresh tag works the same way as hitting the refresh button in the browser. Even if content has a valid expiration date, the browser will ask for validation that it has not changed from the server of origin. This essentially defeats the purpose of setting content expiration dates.

e) Browsing Multiple Pages or Hitting the Back Button

While in the same browser session, all content for a site will be served from the local browser cache. If a user clicks through multiple pages of an application and the same graphics and elements are found on each page, the request will not be sent to the origin web server. Instead it will be served from the local cache. If the user re-visits a page during that session, all of the content—including the HTML—will be retrieved from the local cache.

f) Refreshing Contents

Users might also hit refresh on a page to check for new content, such as an updated sports score or news article. Hitting refresh results in an "If-None-Match" header being sent to the origin web server for all content that is currently on the disk cache, independent of the expiration date of the cached content. This results in a 304 response code for each reusable item that is currently in the browser's cache.

g) New Browser Session

If a new browser session is started and a user returns to a frequently visited site, the local browser cache will be used (based on the browser settings). If a valid expiration date exists for cached content, it will be delivered directly from the cache and no request will be issued to the origin web server.

II. LITERATURE REVIEW/ RELATED WORK

One of the major challenges associated with the Internet is the problem of increased response time caused due to the ever – increasing traffic on the Internet. Many solutions, both hardware and software, have been suggested to overcome this challenge. The popular hardware solutions are to increase the bandwidth of the connection and to replicate the web documents at many locations. Increasing the bandwidth will increase the data transfer rate, and hence decrease the response time. The replication of documents will facilitate the nearest document to be fetched, minimizing the response time. Recently, much research has focused on improving Web performance by reducing the bandwidth consumption and WWW traffic. It means that fewer requests and responses need to go over the network and fewer requests for a server to handle. Despite the fact that there have been great efforts for this purposes the results are not sufficient. The most popular software solution to the problem of increased response time is web caching. Web caching is the technique of locally storing the frequently requested documents so that repeated requests to the same document can be service from the cache itself. instead of fetching the document from the remote server. This will considerably decrease the response time involved. several studies have presented the Web caching as the most beneficial solution for web performance improvement. Web caching systems can lead to significant bandwidth savings, higher content availability, reducing client latency and increasing server's scalability and availability.

III. CONCLUSION

Eliminating the need for the browser to download content on repeat visits can greatly improve the performance of web applications. When a user visits a web page, the contents of that page can be stored in the browser's cache so it doesn't

need to be re-requested and re-downloaded. Efficiently using the browser cache and updated contents of the cache can improve end user response times and reduce bandwidth utilization. There are many factors that impact whether or not content can or will be retrieved from the local browser cache on repeat visits, including the browser settings, the web site, and the user's behavior and content cachebility..

REFERENCES.

[1] White paper Written by Dawn Parzych Acceleration System Architect (ASA)
[2] Web caching: Optimizing for internet and Web Traffic (White paper)
[3] Adaptive web caching algorithms.. Geetika Tewari and Kim Hazelwood.
[4] Evaluation of Efficient Web Caching and Prefetching Technique for Improving the Proxy Server Performance G.N.K.Suresh Babu and S.K.Srivatsav.
[5] Performance Evaluation of Web Proxy Cache Replacement Policies Martin Arlitt, Rich Friedrich
[6] An Expiration Age-Based Document placement Scheme for Co-Operative Web Caching May 2004. IEEE. Transaction on knowledge and Data Engineering. Volume 16, Issue 5.
[7] Evolution Techniques for web caching January 2002.