Web Components: A Concept for Improving Personalization and Reducing User Perceived Latency on the World Wide Web

Christos Bouras
Computer Technology Institute - CTI,
Kolokotroni 3, 26221 Patras, Greece

Agisilaos Konidaris
Computer Technology Institute - CTI,
Kolokotroni 3, 26221 Patras, Greece

Abstract. In this paper we address one of the performance problems of the World Wide Web, known as User Perceived Latency (UPL). We begin by presenting a new definition of UPL. Next we present a concept that may benefit the WWW and reduce UPL. This is the concept of Web Components. We show how Web Components can work together with other concepts for efficient Web Data manipulation and delivery, in order to reduce UPL. We then show how Web Components can also be used to support Web personalization as a means for reducing latency. After that we present the operation of the WWW model (Web Servers, Web Proxies and Web Browsers) for supporting the Web Component concept.

Keywords: User perceived latency, Web personalization, Web data, Web Components

1. Introduction

The growth of the Internet and especially the WWW has brought forward some challenges that were imposed by the tremendous and unforeseen growth itself [1,2,3,10]. It is obvious that with the rapid increase in WWW hosts, the information that has become available on the WWW has also grown substantially. The procedure of efficiently acquiring information from the WWW has become a very popular and important research activity, since the ability to find and explore information that is up to date and actually helpful is considered essential to users.

In GVU's survey of October 1998 [4] users answered to questions concerning their browsing habits and problems. Some interesting statistics that were derived from the survey, are synopsized in the following statements:

1. The basic problems that users encountered were the problems of connection speed, slow advertisements and the acquisition of new information.
2. Users find new information through other web pages and search engines.
3. Most users spend 5-10 minutes searching for information.
4. Users search for specific information on the WWW.

The statistics mentioned above confirm our initial remark that some of the most important challenges in the WWW today are the retrieval of useful information and the reduction of UPL. The motivation behind the introduction of the Web Component (WC) concept lies in the realization of basic WWW problems and challenges. The two basic problems that we intend to address are:

1. The improvement of web personalization by making information available through Web Components
2. The reduction of UPL

These goals are difficult to achieve, without applying fundamental changes to the WWW browsing and data retrieval procedures. On the other hand, due to the size of the WWW and the millions of its users, some radical change to the basics of the WWW, such as HTML, HTTP [6] and TCP/IP, would surely not be welcomed. It is obvious that the most efficient improvements would be those, that would build on existent appreciated and standardized techniques by applying minor changes. This was another one of our goals when creating the concept of Web Components.

The information that we actually need from a web site can many times be "well hidden" among other “useless” information, and may also be poorly linked to. The problems that users encounter when visiting a Web site [4] clearly show that the acquisition of information in the WWW is a difficult procedure. We attempt to solve this problem with the introduction of Web Components that can better organize and clearly depict the "nature" and relevance of information. In other words we attempt to make the WWW more personalized and adapt the provision of information to every user’s needs.

Web data transport (and the resulting latency) is the other major challenge in the WWW. Users consider this to be another major problem in their browsing procedures. Users would like web pages to load instantaneously after clicking on a hyperlink or typing a Web address.

The basic goal of this paper is to introduce an innovative methodology that we have called Web Components. We show that the use of Web Components in Web design can help the Web browsing procedure itself in terms of enabling users to navigate and be orientated while visiting different web sites. Web Components may also be used to reduce the users' perceived Web latency. With the use of techniques that have already been proposed, such as P-HTTP and predictive pre-fetching, we show that Web Components can play a very important role in the reduction of Web latency.

2. Related work

A lot of important work in Web site design has been done by Jakob Nielsen and can be found at [8]. He proposes that grouping and subheadings should be used to break a long list into several smaller units. Jakob Nielsen also urges web designers to provide a starting page that provides an overview and several secondary pages that focus on special topics. He proposes that link titles should be used to provide users with a preview of where each link will take them. In another part of [8] Jakob Nielsen notes that speed must be the overriding design criterion.

The issue of improving data transport has been addressed by many different solutions that apply to clients, servers and proxies.

Persistent HTTP is presented in [5,12,14] and has become a part of the HTTP/1.1 [7] protocol. This technique enables the existence of a TCP connection even after the transfer of the page component that it has been opened for, in order to be used for the transfer of other components. Predictive pre-fetching [15] is another very interesting solution and enables the request and delivery of web pages that have not been actually requested by the user based on a predictive algorithm, determining what the user will request next. The goal of [9] is to describe a Web transport protocol (WebTP). This transport protocol is absolutely client oriented and provides an alternative combination to HTTP and TCP.

The cache related improvements to UPL are also very significant. The first one is hinted caching [13]. In this approach a proxy server is "hinted" on the importance of a resource and decides whether to cache it or not according to its "hinted" significance. The second important cache related improvement that we took into consideration in our work is the concept of caching of "composite objects" [16].

3. User Perceived Latency on the WWW

UPL is a different concept than Web latency. The difference is somewhat "delicate" but exists none the less. In [17] UPL is defined as the period of time, starting at the moment when a user issues a request for a document, till the time a response is received. This is a general definition of UPL that must be analyzed further in this work in order to better target the proposed methods for reducing it. The "delicate" variables that are included in the aforementioned definition are:

1. The time that a user issues a request
2. The time that a response is received

The most controversial factor in the above definition of UPL is the determination of response time. Which time is considered to be the response time? Is it the time when the initial response for a request URL is received at the client or is it the time that the whole URL has been brought to the client? The true value of response time can not be easily determined. Conceptually, the time that a user perceives as the response time of a URL is the time when he begins to understand certain aspects of the page. Since this time is very "user specific" we must use another method to determine it formally. In this paper we will consider that the User Perceived Latency of a web page is the time that a user reacts to it, by clicking on a link in the page of typing another URL. To conclude this part of the analysis we present the general formal definition of UPL as it is going to be used in this paper. The definition is summarized in the following formula:

$$UPL=T_{Search}+(T_{Response}-T_{Request})$$

The above formula dictates that UPL is equal to the time that a user spends searching for specific information on the WWW plus the time calculated by the subtraction of the time of a request from the time of the response (as defined above). The main issue here is the determination of the time that a user spends searching. In this paper we intend to provide a methodology that will reduce this time by introducing a completely client specific browsing methodology.

4. The Web Components concept

In this paper we will refer to three types of components. The first, are the Standard HTML Components (SHC). These components may be inline images, text, hyperlinks, videos e.t.c. In general a SHC is everything that can be embedded in HTML.

The second concept of components, consists of many SHCs grouped together, and we call them Conceptual HTML Components (CHC). A CHC can be considered as a group of SHCs. In general, a CHC may consist of an inline image, several hyperlinks and some text, grouped together. The special characteristic that ties all these SHCs together in a CHC is their reference to information of the same thematic category. As we mentioned above, the concept of CHCs has already been adopted in the design of many Web sites without being officially declared or standardized. In order to make this clear we present a paradigm of a Web site that uses CHCs today. Figure 1 shows the homepage of CNN.com found at http://www.cnn.com/ (7-5-2000 at 5:30+2 p.m.GMT).

In Figure 1 we have designated five CHCs on the Web page as well as some of their attributes.

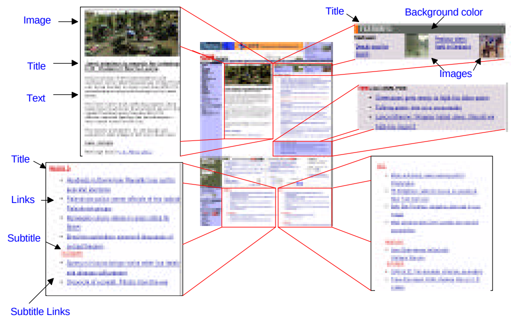

Figure 1 Conceptual HTML components on the CNN homepage.

The third concept of components that we will refer to in this paper is that of Web Components. The definition of Web Components, that will be used throughout this paper from now on is the following:

A Web Component is a conceptual reference to the contents of the WWW (not only specific Web pages or Web sites) that consists of well defined thematic areas on a web page (CHCs) including all linked pages to every CHC. In other words Web Components consist of CHCs and all linked pages to them.

Figure 2 contains a representation of a Web Component.

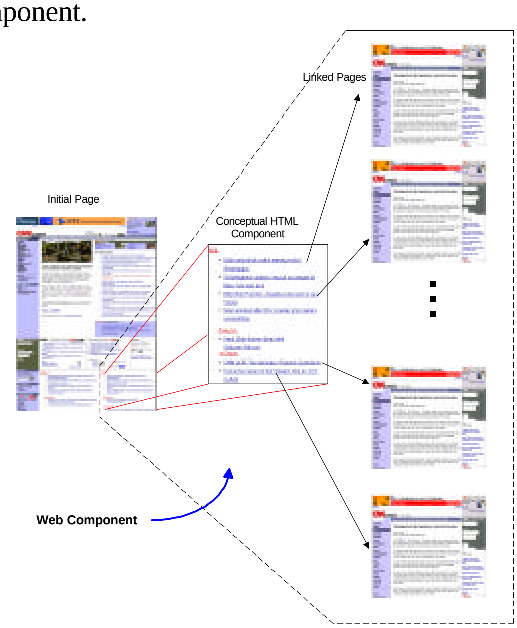

Figure 2 A representation of a Web Component

5. Use of CHCs - proxy log analysis

In order to determine and establish that WWW users, actually make use of the concept of CHCs we conducted an analysis of Proxy server log files that were kindly provided to us by the Proxy service of the University of Patras [18] and the Transparent cache service of GRNET [19]. In order to quantify the use of CHCs we selected sites that could be decomposed into CHCs such as Cnn.com as already mentioned. In general, all sites that fall into the portal category have very "information heavy" index (default or main) pages that may generally be decomposed into CHCs. In our analysis we counted the number of times a CHC was used by all users of a certain site in a period of one week and the number of times that a specific user utilized a specific CHC in consecutive sessions.

In the case of all users we calculated a usage of over 30% of specific (no more than 4) CHCs. This means that over 30% of all hits targeted at a certain web site (not only the index page) are targeted to specific (no more than 4) CHCs of the main (default or index) web site page. In the case of a Greek Portal called In.gr [7] we measured a total of 34% of hits targeted at only two CHCs on their index page. The index page consists of over 15 CHCs.

The case of individual users was also interesting. The results here were also revealing concerning the use of CHCs. We found that users target the same CHC once in every two visits to the same site. This means that there is an almost 50% probability that a user will use a CHC that he has used in a site during previous visits. This result actually means that the user has found useful information there.

The results that we have mentioned above led us to the conclusion that certain CHCs are considered very important by users of most web sites. Thus, it would be useful to eliminate most of the other CHCs in order to provide targeted, updated and useful information very quickly and efficiently on the WWW.

6. WWW personalization

The initial idea that actually led us to introduce the Web Component concept was personalized browsing. Web statistics [11] show that web users visited only 11 unique sites from their home and 27 from work during a browsing session, in April 2000. If we combine this statistic with the observation that most users begin their web surfing with the use of only a few web portals we come to realize that users get into the habit of using specific sites for specific information. Our initial thought was to enable the user to browse the specific site information topics that he selects from every site, all at once. The goal was to let the user select the CHCs that he was interested in from every page, and include them all together in one web page. The user would then only have to request one URL and all the CHCs that he had selected would appear (updated) in the requested page. The idea is very similar to the personalization techniques used in many sites today just enhanced to include the whole WWW. A possible result page is shown in Figure 3.

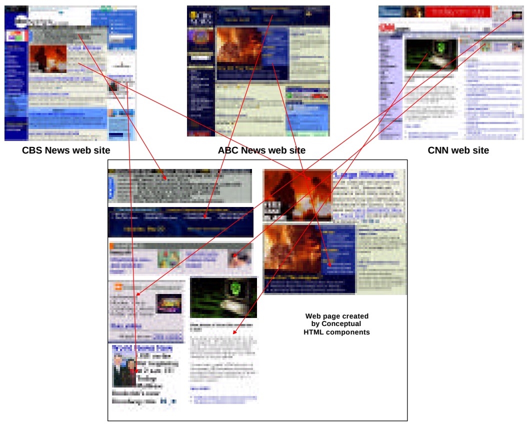

Figure 3 How a Web page may be created by Conceptual HTML components

7. Internet model that may support WCs

The client-server model can be used to support the WC concept. The approach that we will describe here is fully client sided support of WCs.

The idea of this support is to enable the browser that resides on the client to request WCs. The normal HTTP/1.0 procedure is to request a page/file (Resource in general) from a Web server and wait for the server to respond. In our case the client must be able to request a CHC from a web server. This can not be done directly with the use of the HTTP/1.0 protocol since parts of pages can not be requested separately from a server. It is actually impossible to request, with the use of normal HTTP/1.0 procedures, only a part of a web page. In the case of HTTP/1.1 though the possibility to request a part of a web page is made possible through the range requests. This procedure is not very useful to us since one has to know the bytes of data that must be requested in order for them to include the component in question. The best case scenario would be for the server to be able to send (after sending all headers) only a part of a Web page. In order to be able to do so, the server would have to be informed of the portion (CHC) of the page that is requested by a client. This would require a field inside the request that would define the CHC in some way. After receiving this request, the server should be able to parse a Web page and send only the requested part through the network

It is obvious that the server sided procedure described above requires some alterations in the implementation of the HTTP/1.1 protocol. The results of this alteration would be multidimensional. If the HTTP/1.1 protocol was tuned to be able to accommodate CHCs, the WWW browsing procedure would be completely changed. The benefits of embracing a scheme such as CHC are significant and provide answers to many WWW related problems.

A very simple alternative solution would be to fetch the html files describing different web pages and then enable the browser to show only the parts of those pages that the user has selected. This approach is not as efficient as the full server sided support of CHCs but none the less it is a step forward compared to the current browsing model.

8. Proposed changes to HTTP/1.1

Studies have shown that HTTP/1.1 benefits a user at about 20% in the speed up of web page loading, over HTTP/1.0. The question of whether the employment of HTTP/1.1 is worth the time and money that is going to be spent on migration, is one that has not yet been undoubtedly answered. We believe that HTTP/1.1 introduces many improvements over HTTP/1.0. The goal of a Hypertext Transfer protocol is not only to optimize network traffic but also to provide "satisfaction" to the user. If users had no problems with network speed and performance we would not be researching protocol improvements. All the scientific work in the field of the improvement of network protocols, ultimately aims at maximizing "user satisfaction".

Conceptually, if one was able to improve user browsing satisfaction, without altering network bandwidth conditions or web content provision he could claim that he had improved network protocols. In our approach we have chosen to leave bandwidth and content the same but alter protocol, and web design concepts. In terms of protocols we propose minor changes to HTTP/1.1, which has many "hidden" virtues. In terms of web design we only propose a certain "organization" of web content without imposing changes to the actual content.

The changes to HTTP/1.1 that we propose must be applied to both clients and servers that use the protocol. They both have to do with the ability of HTTP/1.1 to request and respond in terms of CHC and not in the classical manner. In order to make the whole process a bit easier we provide a scheme that only involves changes to the client. The steps in the process are the following:

1. The client requests a web page
2. The server sends the html text file describing the page
3. The client parses the file and determines the components needed by the user (selected during another browsing session or based on certain personalization parameters)
4. The client then eliminates the overhead components from the text file
5. The client requests the SHC that are left inside the new html text file

In this case the client must request the html text files from all sites involved in the page created by the user. After determining the components that are needed from each of these files, the client must be able to combine these components in order for them to be shown to the user inside one web page. After the selection and combination processes are over, the client creates a new text file and starts requesting the SHC's needed to display it. After that, the page is displayed to the user. The only open issues that remain, are the issues of which URL the client will request, to get the personalized web page and how the user will be able to choose components in the first place. In order to clear up the whole process we describe a possible scenario where we have assumed that the user's browser is CHC enabled.

A user is browsing the WWW and requests a URL. The algorithm that is executed is the following.

```
Check if URL requested is virtual or real
If virtual then
   Open file containing Real URLs and CHC Reference
   Numbers
      Do while not EOF
         CurrentLine=Read Line
            If CurrentLine is real URL then
               Request text file from real URL
                  Do while there are CHCs in Text File
                     Select CurrentCHC
                  loop
            End if
      Loop
   FinaltextFile=Combine all Text Files
   Request all CHCs contained in FinalTextFile
   Show page to user
End if

If Real then
      Request Text file from server
      Parse text file and distinguish CHCs
         Do while there are CHCs
            Request All files contained in CHC
            Show CHC
         loop
End if
```

In the algorithm described above, only the HTTP/1.1 client has to be aware of the CHC concept. The whole procedure is transparent to the server and the intermediate network elements such as proxy servers. In this case the network is relieved from having to transfer unwanted information contained in certain server files. The implementation that we have proposed here is quite simple, but may be considered as an initial Web Component approach. The network has been relieved from unwanted information transfers but the initial text files (containing HTML) that are always requested, also contain text information that is eventually not used by a user. The way to get around this deficiency is to enable the server to send "chunks" of information that comprise of CHCs (server sided approach).

The above algorithm proposes that a text file (or any other internal browser information storage facility) is used to store the information that has to be known in order for the browser to request CHCs. The information is inserted into the text file after selections of CHCs by the users during browsing sessions. This clearly shows that the browser must also provide a CHC selection process to users during their browsing sessions. The information stored in the text file comprises of the actual URLs in the WWW, where an updated version of each CHC can be found and a reference number of the CHC in the actual text page, showing the order of appearance of the CHC in the actual Web page. The reference number is determined for each CHC during a previous parsing of the Web page by the client's browser. For every URL contained in the locally stored text file, the browser requests the corresponding HTML file from the WWW and parses it to find the CHC corresponding to its reference number. After executing the same procedure for all URLs in the locally stored text file the browser combines all the resulting CHCs into one html file and presents it to the user.

The algorithm that has been described above proposes the use of what we call virtual URLs. Virtual URLs are actually non existent on the WWW. These URLs are pointers to the locally stored text files containing the real URLs containing CHCs and their reference numbers. When a user types a virtual URL (or selects it from the bookmarks) the aforementioned algorithm is executed and the derived web page (containing all previously selected CHCs) is shown.

In order to keep the consistency of the information that will be presented to the user, the client opens a persistent connection for every CHC in order for it to be transported to the client.

After requesting a Virtual URL and all the CHCs contained in the page are loaded, a predictive algorithm is executed by the client's browser, in order to predict the URL that the user is going to request next. The predictive algorithm takes into account the already requested URLs and the significance number given to the referred resource by the administrator. After the execution of the algorithm all referenced CHCs are prioritized, and their prefetching begins.

9. Conclusions

In this paper we have described a concept that may apply to every module of the WWW data transport model: clients, servers and proxies. The application of the WC concept to WWW servers will improve server load. The inherent advantage of WCs is that they relieve WWW servers from serving files that are not actually needed by users thus, saving bandwidth. This work attempts to view web page objects as related, rather than unique entities according to the message that these objects wish to convey to WWW users. The concept that we have presented here provides users with the power to personalize the WWW and view only specific information that they have selected. The work presented in this paper proposes the improvement of Web personalization and the reduction of user perceived latency through the extension of existent protocols. The concept may be imposed and implemented in the future through various techniques most prevalent of which is XML.

The most important feature of our proposal is that the power of web navigation is transferred to the WWW client thus, the overall navigation procedure is enhanced.

10. Future work

Our future work will focus on the integration of the Web Components concept in HTTP/1.1. We intend to make the protocol Web Component enabled by changing its "resource" concept. At the moment the protocol supports only range requests in terms of component querying. In order for the protocol to be able to inherently support the Web Components concept every component must be referenced as a unique resource. We have already implemented a mechanism that supports this feature in both Web servers and browsers but it is not integrated in the HTTP/1.1 protocol. This mechanism may be viewed as a protocol filtering procedure that shells HTTP/1.1 giving the overall notion of the support of Web Components. In the future we intend to integrate this mechanism into the protocol, thus enhancing it.

11. References

[1] Internet Software Consortium, http://www.isc.org, 2000.

[2] Netcraft Web Server survey, http://www.netcraft.co.uk/Survey/, 2000

[3] Internet Statistics: Web growth, Internet growth, http://www.mit.edu/people/mkgray/net/, 2000

[4] GVU's 10th WWW Survey results, http://www.gvu.gatech.edu/user_surveys/survey-1998-10/, Copyright 1994-1998 Georgia Tech Research Corporation. All rights Reserved.

[5] Venkata N. Padmanabhan "Addressing the challenges of Web Data Transport" PhD Dissertation , University of California at Berkeley, 1998

[6] T. Berners-Lee, R. Fieldings and H. Frystyk, Hypertext Transfer Protocol-HTTP/1.0. RFC, May 1996, RFC-1945

[7] In.gr Portal http://www.in.gr

[8] useit.com: Jakob Nielsen's Website, found at http://www.useit.com/

[9] Rajarshi Gupta, "WebTP: A user-Centric Receiver-Driven Web Transport Protocol", M.S. Thesis, Fall 1998

[10] CyberAtlas: Internet Statistics and Market Research for Web Marketers, found at http://cyberatlas.internet.com/

[11] J. Gettys and A. Freier, HTTP Connection Management, Internet Draft, IETF, March 1997, Expires Sep. 1997

[12] Jeffrey C. Mogul, The case for Persistent-Connection HTTP, Western Research Laboratory Research Report 95/4

[13] Jeffrey C. Mogul, Hinted caching in the Web, 1996 SIGOPS European Workshop

[14] Venkata N. Padmanabhan, Jeffrey C. Mogul, Improving HTTP latency, Proc. Second International WWW conference, October 1994

[15] V. N. Padmanabhan and J. C. Mogul, Using Predictive Prefetching to Improve World Wide Web Latency, ACM SIGCOMM Computer Communication Review, Vol. 26, No. 3, July 1996

[16] Craig E. Wills, Mikhail Mikhailov, Studying the impact of more complete server information on web caching, 5th International Web caching and Content delivery Workshop, Lisbon, Portugal, 22-24 May 2000

[17] Edith Cohen, Haim Kaplan, Prefetching the means for document transfer: A new approach for reducing Web Latency

[18] Proxy server of the University of Patras http://www.noc.upatras.gr/proxy-traffic/index.html

[19] Transparent cache service of GRNET http://www.grnet.gr/proxy/index_en.html