Web Browser Intelligence: Opening Up the Web

Rob Barrett  Paul P. Maglio  Daniel C. Kellem
IBM Almaden Research Center
650 Harry Rd.
San Jose CA 95120
{barrett,pmaglio,kellem}@almaden.ibm.com

Abstract

The world-wide web has brought us two important resources: ubiquitous browsers and a global information repository. In the current model, these two resources are tightly coupled. By introducing the concept of a programmable intermediary between browser and server, Web Browser Intelligence (WBI) has released this coupling. WBI allows the browser to execute arbitrarily complex commands using the network of web servers as an information resource. This architecture enables automatic personalization of the web for individual users, automatic restructuring of information on servers, and collaboration among web users.

The world-wide web consists of two tightly-coupled technologies, browsers and servers. Web browsers are cross-platform, multimedia information displays and command generators. Commands originate with users as hyperlink clicks, manually-entered URLs, and filled-out forms. Web servers work together as a network-based, distributed information system with a simple protocol and simple URL-document indexing. Together, browsers and servers conspire to form a collaborative information system on a global scale. Nevertheless, we believe that the tight coupling between browsers and servers has undesirable consequences. Notably, every user sees the same view of the web because the displays generated by the browsers are almost completely determined by the servers. This results in the impersonal presentation information that many find unattractive and difficult to use.

We extend the web model by introducing programmable intermediaries between browsers and servers. Our system, called Web Browser Intelligence or WBI (pronounced “Webby”, and available at [1]), receives commands from a browser and returns documents for the browser to view. WBI interprets a browser command and then dynamically constructs a response using web servers or other resources to access information, possibly storing and accessing user and community models in the process. WBI is an extension of the simple web proxy, which is used to cache information and forward requests to appropriate servers [2]. WBI can perform the simple proxy function, but it can also perform arbitrarily complex actions as a result of browser commands.

One example of the utility of this system is our personal history module, which enhances personal productivity on the web. A user accesses the web in the usual way, but every access goes through WBI, which records a history of all URLs visited and the contents of all of the pages viewed. This stored information allows WBI to dynamically generate hot lists of popular pages and to recommend shortcut links. The user can also query the personal history by accessing special WBI pages. This personal history allows the user to search through the contents of previously viewed pages by using keyword searches or by reviewing previously travelled paths. WBI also modifies the appearance of web pages by editing the HTML that makes up the pages. For example, WBI adds colored lights around links to indicate the speed of the network connection to the server. These “web traffic lights” warn users about overloaded or inaccessible servers. The personal history module is simply one example of the added function that can be obtained by introducing an intermediary.

WBI modules are defined by a set of monitors, editors, generators, and rules. A monitor is given a copy of a web request and the resulting page so that it can record user actions. For example, our personal history module uses monitors to record the series of URLs visited and the contents of the pages viewed. An editor is allowed to modify the resulting page before it is returned to the browser. For example, an editor is used to add shortcut links to a page or extra buttons for accessing WBI functions. A generator is used to handle web requests. For example, a generator is used to search the personal history and return a list of pages that contain a given keyword. Rules are used to define which specific monitors, editors and generators should be instantiated for a given request. For example, certain editors only
wish to modify HTML documents. Some monitors may only want to observe requests made to servers in a certain domain. A more complete description of the WBI architecture can be found elsewhere [3, 4].

In conclusion, we have found WBI to provide a flexible architecture for expanding the web paradigm. We have added function to browsers to increase productivity. We have also added function to servers by intercepting requests and automatically adding content based on other information on the server and past usage models. We have also begun to add collaborative capability to the web through intermediaries communicating with each other. By adding this flexible intermediary model to the web paradigm, we have opened a pathway for new web functionality.

References


