Temporal HTML

Lina Liu
Computer Science Department
University of Victoria

Although HTML serves many applications very well as a high-level authoring language for the World Wide Web, it provides no temporal support for a site. With rapid development of web technology nowadays and the appearance of more and more time critical applications on web, it is not sufficient anymore to present static information. Making a site “dynamic” becomes more and more important.

The nontemporal (static) nature of conventional HTML is a serious hindrance to a dynamic World Wide Web. Some of the problems arising are:

1. Web servers often have to retrieve archived material, in addition to providing the most updated version. For many kinds of sites - newspaper sites, stock market sites, the retrieval of a certain version by date is a must functionality. However, the existing HTML only directly supports keep one version in the system.

2. In those time critical web applications, such as the stock market, the information viewed at one moment might become stale in the next moment. The existing HTML does not provide a mechanism to synchronize the server and clients, so that client will be informed the change if any when they are being viewed.

3. Web authors feel more and more the pressure to plan a site dynamically rather than statically. They not only want to plan what is to be posted, but when it is to be posted. Existing web server does not provide for the selection of versions of a file according to a requested time the author has to manually update the site to let it appear dynamic from the client perspective.

Companies such as Netscape and Micros oft are devoting much to extending the capabilities of the Web. As a result, lots of HTML enhancements appear, such as
JavaScript, DHTML, XML etc.

Although they make the web more interactive, they don't really address the problems just cited.

For example, XML provides developers their own special purpose tags that can be defined for a variety of purposes and allow HTML to convey complex mathematical expressions. DHTML extends HTML for presenting multimedia applications that run on the user’s desktop without the server’s interaction, to present interactive databases and documents. None of them solve temporal demands of a web in a natural way and provide efficient mechanisms to implement the complicated temporal phenomena a web application may require.

THTML use temporal logic to provide efficient solutions for most temporal demands of a web site and has a great potential for representing complicate time-critical web. THTML has the following advantages over HTML:

1. THTML provides a mechanism to allow one to define arbitrary versions for the same HTML file. It also allows one to define versions for parts of a page.
2. THTML also allows the user to define temporal versions by defining a timestamp for each version, which indicates the valid time to post on the Web.
3. THTML allows a client to define a time in an HTML file—time criteria, which allows client to retrieve a certain temporal version by the time point. The time can be a point relative to “now”, such as yesterday or the day before yesterday, or an absolute time point such as May 3rd, 1998 12:12:12.

1.1.3 THTML Solution

THTML introduces DTL, a discrete temporal logic formalism, to provide temporal support, and intensional logic to provide a version control mechanism on top of HTML. THTML provides a kind of temporal language by which the user can define most
common time patterns—time period or time point, infinite time or recurrent time (say, every Monday and Tuesday.)

Each client request to the server is accompanied by a time-of-interest. Then the server will find temporal version whose range of validity best fits the client's time-of-interest. The response is configured dynamically, and discarded once sent.

The dynamic configuration of THTHTML allows the user to define an arbitrary version of each component of a site.

By defining the temporal versions for each part, we can greatly increase the number of different available versions of the whole. A file of a certain version is composed of the best fit of corresponding version of each part. If no best fit is found, a generic version will be used. By keeping the versions of each part and the dynamic configuration of a page, much space can be saved compared with saving versions of each page and the maintenance is also avoided for each version.

The working mechanism can be illustrated by an example below:

A newspaper site includes title, the banner, body and footer. The title and footer are the same in every day. While the body is different every day, the title, banner and footer are constant.

We can divide the page into three parts, title, banner, body and footer, for we can define only one version for title, banner and footer but many temporal versions for body parts.

If the server receives a request for yesterday's page, the best-fit version for each part will be retrieved. This means the answer to the request is the result of assembling the version of the body associated with yesterday's date, together with the generic (time invariant) title, banner and footer.

THTHTML also provides a kind of synchronization between client and server by “push” technology incorporated at server site. When the client is viewing a file, if it is stale—not the best fit in the next moment, the server will inform the client by a dialog box popped up at the user-defined frequency saying the update and if the client would like to update the page or not. If the client clicks yes, the page will be updated, if no, the stale page will be kept. The same mechanism is adopted for the previous visited pages
when they were updated

THTML was implemented using the same techniques (and some of the source) used for IHTML. One great advantage is that it inherits the (non-temporal) versioning facilities of IHTML. This means, for example, that we can have different versions of a time-changing page, or have a page with programmed changes of versions.