ENHANCEMENTS TO TIME.GOV: THE NATIONAL WEB CLOCK FOR THE UNITED STATES

Andrew N. Novick
National Institute of Standards and Technology
325 Broadway, Boulder, CO 80305, USA
Tel: (303) 497-3378; E-mail: *novick@nist.gov*

Samuel Ou
United States Naval Observatory
3450 Massachusetts Avenue, NW
Washington, D.C. 20392, USA

Abstract

Since 1999, the official time for the United States has been presented with a graphical interface on the World Wide Web. Major improvements to the site, www.time.gov, will soon be implemented. The goal was to continue to display an animated clock with an estimated path delay from the source, but to improve upon the efficiency and functionality of the Web site.

Switching the interface from the Java/Perl environments to using Flash and XML has improved the efficiency on the server side and increased ease of use for the variety of clients and firewall settings. Several options have been added, including an audio "tick", 12 or 24-hour digital clock display as well as an animated clock face. International time zones with Daylight Saving Time corrections are also represented. This paper includes an explanation of how the site works, as well as traffic and load issues.

THE PREVIOUS VERSION

The Perl/Java version of *http://www.time.gov* is the product of a collaboration of the National Institute of Standards and Technology (NIST) and the United States Naval Observatory (USNO), both of which are official sources of time for the United States **[1]**. The site works by first sending the client a Hypertext Markup Language (HTML) page displaying a map of the United States with areas of alternating colors indicating different time zones **[2]**. When a user clicks on a time zone, several parameters are sent to the Web server where a Common Gateway Interface (CGI) script written in Perl is executed by the Perl interpreter on the server. The script gets the current time and date from the server clock, which is synchronized to the official United States time **[3]**. After the code determines whether it is Daylight Saving Time (DST) or Standard Time (ST), a Java applet is sent to the browser. The client is required to have the Java Runtime Environment installed, which allows the applet to query the server for the time and display an animated numeric (digital) representation of the time. A non-standard port (8013) is used for the request. Using the client's computer clock as a timer, the delay of the process is measured and displayed. HTML code is sent to the browser, which displays additional text and links. There is some

embedded Javascript code to calculate where the Sun is above the horizon ("daytime") and below the horizon ("nighttime"). This code requests a gray-line map image from the server, based on the current position of the Sun.

After the initial synchronization, the "running" clock uses the client's computer clock as its oscillator, and periodically refreshes itself with the server time. For this reason, it is meant to be used only as a time of day source, not for timing applications or calibrations. Each refresh initiates a call to the Perl interpreter on the server.

Although Java has fairly good penetration with users of the Internet, the *time.gov* site has not worked for everyone. Based on e-mails received from users, the animated clock does not always work, and some people and must use a non-Java version of the page which displays a static "snapshot" of the time. The animated clock periodically stops working for many other clients (using mostly the Windows operating system), and Java must be re-downloaded in order for it to work again. Also, with the increasing emphasis on Internet security, many clients and network administrators implement strict firewalls, which sometimes block port 8013, which the applet uses to get the timestamp from the server.

THE NEW VERSION

The new implementation of displaying the official time in a Web browser uses the Adobe Flash Player* plugin, which has very high penetration on personal computers as well as other Internet-enabled devices. Also, it gets the time information from the server using port 80, which is the default port for Hypertext Transfer Protocol (HTTP) requests. By creating a site that can be accessed by more Web-based clients, we anticipate that the official Web clock will be more widely used and appreciated by the general public.

The new site (Figure 1) opens with a page that shows the current time in all of the time zones in the United States at once, so no mouse clicks are required before viewing the time. A user can easily see the current time in several time zones at once, with both digital and analog running clocks. There is a larger, "master" clock that initially shows the time in the user's time zone, if that information can be gained from the browser settings. When any one of the time zone clocks is clicked, the master clock changes to display that local time. A static, non-flash, "snapshot" digital clock will also be available.

Since some states have areas in more than one time zone (usually divided by county borders), there is an option to type in a zip code, which changes the master clock to display the local time in that region. There are also options for 12-hour and 24-hour representations of the time, as well as an option to hear an on-time "tick."

A similar gray-line map showing the current position of the Sun over the Earth is also displayed, which is a very popular feature of the original site. In addition, the map can be expanded to full size with an overlay of the master clock showing the current time for a zone of the client's choice (Figure 2). This "view" makes a visually interesting (and highly requested) page to leave showing when the computer is not being actively used. The page refreshes itself with the server periodically (every 10 minutes), so that the time and map displays are correct.

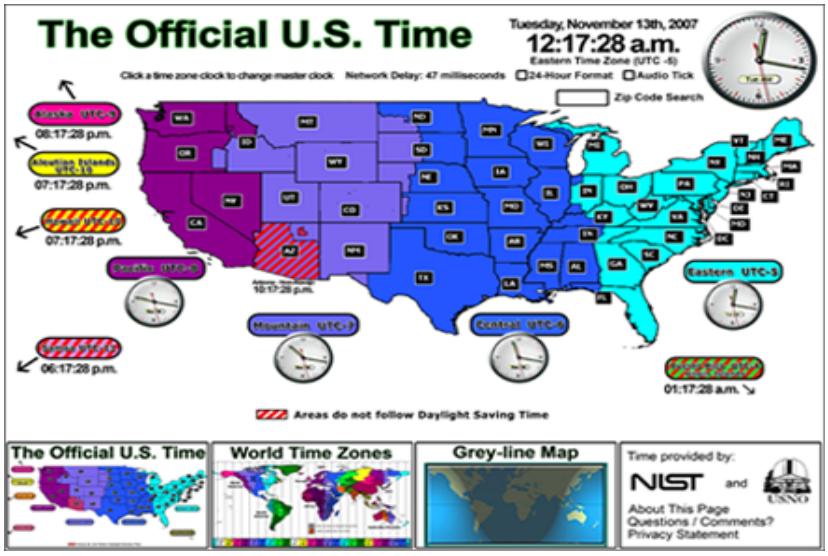

Figure 1. The initial view for the new *time.gov* site.

Figure 2. The gray-line map view.

Another option is the world time zone display. It shows a world map with color-coded time zones and the current time in each (Figure 3). This is another option that is highly requested by users of the current site, since many people know the hour differences between time zones in the continental United States, but they do not know the differences for time zones in other parts of the world. Daylight Saving Time (DST) rules and borders change fairly often throughout the world, so this page uses an Extensible Markup Language (XML) database of time zone rules to display the local times in several countries. This database is kept current in accordance with the *World Time Database Subscription Service**, a paid service that prides itself in keeping as up-to-date time zone information as possible. The site has a

disclaimer that this is not official information and is provided only as a public service and is not legally binding.

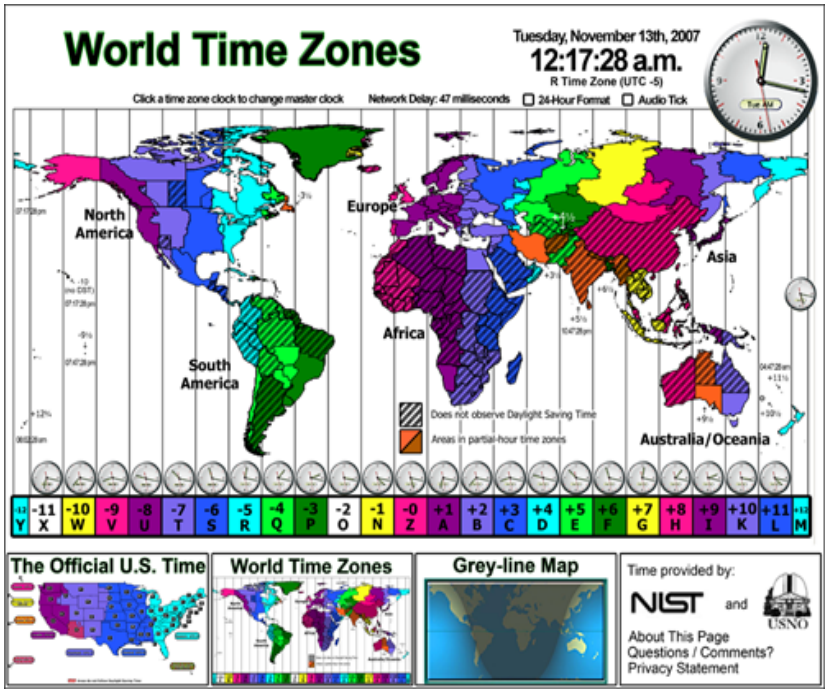

Figure 3. The world time zone view.

HOW IT WORKS

The Flash application is not written with standard Macromedia Flash* tools. Instead, an open-source platform called *OpenLaszlo**, with its own code language, called LZX, is used. LZX is an object-oriented markup language that includes XML tags and embedded Javascript, and it compiles into the *.swf* format to be executed by the Flash Player plugin. It is designed to develop real-time interaction with animations and data retrieval **[4]**.

OpenLaszlo uses different *views* in the interface, and a user can interact (mouseover or click) in the different views to somewhat morph or glide into another view and back, so there are not really set pages between which to surf "forward" and "back." There are universal objects that, once created, can be placed in different views with different parameters. For instance, once an analog clock object has been created with code, it can be referenced multiple times with different parameters, such as the current time in a particular time zone.

When a client reaches the Web site, the compiled *.swf* file is downloaded into the browser and "plays" in the Flash Player. For the Official U.S Time view, the program requests a timestamp from the server. It then cycles through a database of time zones and invokes a clock object for each unique zone with corrections for the time zone and DST (when applicable). The clock objects are overlaid with a time zone map graphic. It also calculates and displays the gray-line map, as in the previous version. There are several dynamic parts of the view, such as clicking on the clocks, entering a zip code, and toggling the 12-24 hour clock and audio on or off.

Another view can be selected from the boxes along the bottom, making it the main view. For instance, if the World Time Zones view is selected, the world map zooms to full screen, and several clock objects are placed on it, getting their offset data from an XML database.

The retrieval of the official time is similar to that in the previous version. The client's computer clock is recorded (to the millisecond) and then a small program called *gettime.cgi* on the server is called though port 80. It is a CGI script programmed in C, which uses a call from the *timeb* library to obtain the time on the server. It is already compiled, so the server has only to execute it when called. It returns a timestamp to the client, which is the number of milliseconds since 00:00:00 UTC, January 1, 1970. The local computer clock is read again and compared to the previous time on the machine. This difference is the delay of the call to the server and is displayed to the client as "Network Delay."

The difference between received timestamp and the client's computer clock is saved in a variable called *RealTimeDif*. In order to display an animated clock, the application displays the running clock of client's computer, offset by the value of *RealTimeDif*. So it does not matter how far off the local clock is for the correct time to be shown. At the top of each second, the corrected time is propagated to all of the clocks in the display (with 1-second resolution) with the correct hour-offset for each. At the same time, an audio object called *tick* is invoked if the audio option is turned on. The *tick* originates from an audio file called *tick.mp3*, but is compiled into the *.swf*, so it does not have to be loaded separately from the application.

An algorithm was created to designate the time zone from a given United States Postal Service zip code. It takes advantage of the range of zip codes for each state. Since there are several states located in two time zones, the affected zip codes were determined and given exceptions. The code is more efficient than having to implement a database entry for each zip code and its respective time zone. However, there are certain disadvantages to this technique. Any changes to zip code or time zone boundary lines will need to be updated. Also, some of the zip code borders do not follow the time zone borders exactly. For these zip codes, the time zone that covers the majority of the area was chosen.

DIFFERENCES ON THE SERVER SIDE

The original site requires a fair amount of processing by the Web server. For each request, a separate Perl interpreter on the server is invoked to run the *.cgi* script. During periods of high traffic, such as on the DST transition days, the Web server receives many simultaneous requests. Also, there are at least two calls to the server clock: one by the *.cgi* and one or more by the applet. There are several files called by each user: the *.cgi* file, three *.html* files, three *java.class* files, and multiple *.jpg* files.

The new version sends an HTML page to the browser, which calls the compiled LZX if the Flash Player is installed. This is a single *.swf* file that includes everything but the gray-line image, and it calls the server clock only once. The server runs the compiled C program, which executes very quickly. When the user changes a view, no files are downloaded from the server, although there may be an additional request of the XML database. The program periodically refreshes the time from the server, but this requires only a call to the C program.

Using load-testing software, we compared between the Perl/Java version to the LZX version. The software generates a number simultaneous requests and records the round-trip response time. In this case, the delay measured is primarily the time of the server response, since the requesting machine and server were both on the same network at NIST in Boulder, CO. The number of simultaneous requests ramped up to 100 over the course of 1 minute. The server handled the load with the Perl/Java version with a

response time of less than 0.5 s until there were about 70 simultaneous requests. At that point it slowed greatly, which would produce very undesirable results for a time service. The server load for the LZX version never generated a response time greater than 0.05 s, and the response time didn't increase significantly with a very high number of simultaneous requests.

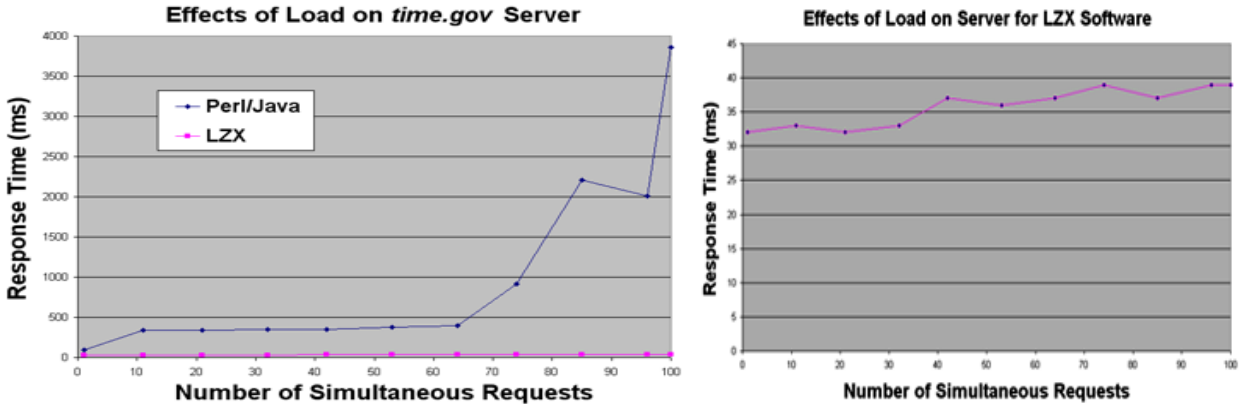

Figure 4. Graphs of load tests.

TRAFFIC

To measure Web traffic statistics, each request of the *.cgi* file is counted as one time request. The current traffic on *time.gov* has grown from around 2 million time requests per month in 2001 to an average of 4.5 million requests per month in 2007. On the dates of Daylight Saving Time changes, the traffic increases greatly, to about seven times the daily average. On 4 November 2007, the site had over 1.1 million time requests, handling as many as 40 requests per second.

MEASUREMENT UNCERTAINTIES

The goal of the site is to provide clients with accurate time of day within 1 s of UTC (NIST) and UTC (USNO). Results of ongoing comparisons between these two time scales are published monthly, and they are routinely within 50 ns of each other **[5]**. So, in terms of milliseconds, NIST and USNO are equivalent, and either laboratory can be used as the official source. The measured round-trip delay is displayed with a resolution of 100 ms (0.1 s). This shows the user the maximum possible offset of the time shown on the page. The measurement does not show where the delays occur in the network path, so the present philosophy is not to remove the delay. For example, if most of the delay occurs before the timestamp is requested and one half of the round-trip delay is removed, too large of a correction would be made and the time shown could actually be *ahead* of the correct time.

In order for a time source to be traceable, its measurement uncertainty must always be known. The Web site *www.time.gov* makes only periodic synchronizations with an official time source (with a coarse measured delay) and then uses the computer clock (unknown uncertainty) as its oscillator. Therefore, the site should not be used for calibrations or measurements of time or time interval.

CONCLUSION

Major improvements have been made to the national Web clock. These changes are expected to benefit the users of the site with more options and better functionality, accessibility, and user experience. The optimization of the software and techniques used should cause much less stress on the servers, allowing the site to work consistently, even under heavy load. Also, this will allow the site to handle many more users in the future.

Before going online as the sole implementation of *www.time.gov*, the LZX version must be extensively tested for functionality and accuracy across different platforms and browsers. Also, it must be approved by the various government agencies involved. It will be also beta-tested by current users of *www.time.gov* for functionality and usefulness. After going online, we expect it to continue to evolve and improve in future years.

** The use of commercial company or product names is for technical completeness only and does not imply an endorsement by NIST or USNO.*

REFERENCES

[1] "*America Competes Act*," United States Public Law 110-069, 110th Congress (9 August 2007).

[2] A. N. Novick, P. R. Franchois, M. A. Lombardi, and W. D. Lee, 2002, "*Time Distribution using the World Wide Web*," presented at the Measurement Science Conference, 23-25 January 2002, Anaheim, California, USA.

[3] J. Levine, 1999, "*Time Synchronization over the Internet Using an Adaptive Frequency-Locked Loop*," **IEEE Transactions on Ultrasonics, Ferroelectronics and Frequency Control, IM-46,** 888-896.

[4] Laszlo Systems, Inc., 2006, "*OpenLaszlo, An open Architecture Framework for Advanced Ajax Applications*," White Paper, *http://www.laszlosystems.com*, November 2006.

[5] National Institute of Standards and Technology, "*NIST Time Scale Data Archive*," *http://tf.nist.gov/timefreq/pubs/bulletin/nistusno.htm*