# Easy Web Interfaces to IDL Code for NSTX Data Analysis

W. M. Davis[a]*

[a]*Princeton Plasma Physics Laboratory, P.O. Box 451, Princeton, NJ, 08543, USA*

## Abstract

Reusing code is a well-known Software Engineering practice to substantially increase the efficiency of code production, as well as to reduce errors and debugging time. A variety of "Web Tools" for the analysis and display of raw and analyzed physics data are in use on NSTX [1], and new ones can be produced quickly from existing IDL [2] code. A Web Tool with only a few inputs, and which calls an IDL routine written in the proper style, can be created in less than an hour; more typical Web Tools with dozens of inputs, and the need for some adaptation of existing IDL code, can be working in a day or so. Efficiency is also increased for users of Web Tools because of the familiar interface of the web browser, and not needing X-windows, or accounts and passwords, when used within our firewall. Web Tools were adapted for use by PPPL physicists accessing EAST data stored in MDSplus with only a few man-weeks of effort; adapting to additional sites should now be even easier. An overview of Web Tools in use on NSTX, and a list of the most useful features, is also presented.

*Keywords: IDL, Web, NSTX, MDSplus*

## 1. Introduction

The National Spherical Torus Experiment (NSTX), like most large-scale physics experiments, depends on custom-written software to optimize the running of the experiment and the interpretation of results. There are over 60 diagnostic subsystems on NSTX, which take around 2.5 gigabytes (GB) of signal data and several more GB of Fast 2-D Camera data every pulse. NSTX takes around 40 pulses per day for 80-100 days per year. There is over 10 years of past data on disk, as well. Custom-written software is time-consuming and expensive to produce, so the use of existing software, and increasing efficiency when producing customized functions, are beneficial. Likewise, users need intuitive, or at least familiar, interfaces to software, especially when coming from other institutions for brief collaborations. NSTX's use of collaboratively-produced software frameworks, such as MDSplus [3,4] and EPICS [5], have helped greatly in these areas [6]. The use of Web Tools [7] on NSTX for the past 10 years has substantially reduced development and learning-curve costs, as well.

* Corresponding author. E-mail: bdavis@pppl.gov, Phone: (609)-243-2546, Fax: (609)-243-3086

*Notice: This manuscript has been authored by Princeton University under Contract Number DE-AC02-09CH11466 with the U.S. Department of Energy. The publisher, by accepting the article for publication acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes.*

## 2. Web Tools in use on NSTX

Dozens of Web Tools are in use on NSTX. Most were initially created as interfaces to IDL code already written for analyzing and displaying NSTX data. Currently, NSTX Web Tools can, among other things:

- Plot or list results of NSTX Tree Data Interface (TDI) expressions or IDL routines, overlaying different shots, with automatic time-scale conversion when signal math is performed.
- Plot from "Scope" [8] files, overlaying different shots.
- Create Movies of 2 fast 2-D cameras, with MDSplus signals vs. time overlaid.
- Query various NSTX SQL databases, for various data ranges, strings, etc.
- Query MDSplus trees for node names, contents of text nodes, ranges of signals, etc.
- Plot Color Contours of EFIT flux [9], with diagnostic sight lines, Thomson Scattering profiles, and other arbitrary signals vs. time.
- List EFIT, LRDfit [10], and TRANSP [11] runs available for a shot by owner.

## 3. Steps to create a Web Tool from existing IDL code

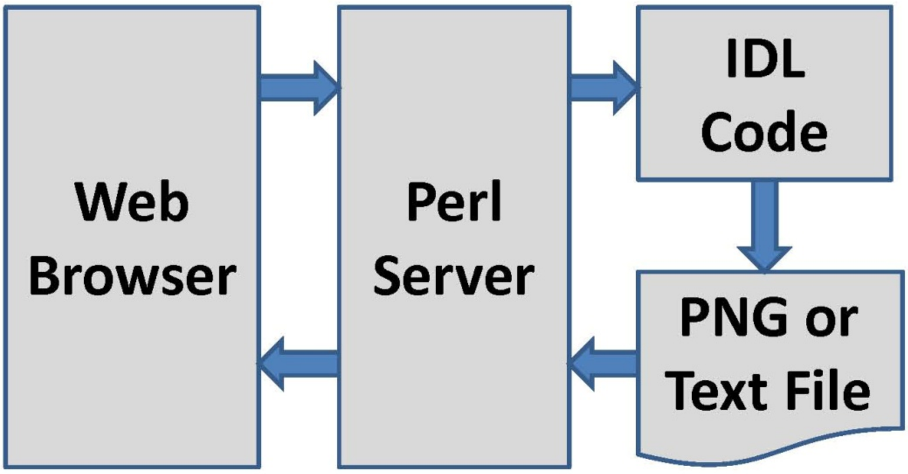

Fig. 1. Components necessary for a Web Tool to call IDL code. For a new Web Tool the Perl Server does not have to be changed. The IDL code must take all its inputs from keywords.

As shown in Fig. 1, three components are necessary for a Web Tool to call IDL code. Custom code is only needed for 1) the user interface web page to be displayed in any Web Browser, and 2) the called IDL code (unless it already exists). The Perl Server script needs no modification. The user interface is usually written as an HTML form with various fields, pull-down menus, check boxes, etc. Many Web Tools on NSTX use cookies and JavaScript, but they are not necessary. Input and selection fields on the Web Tool form must have a name that corresponds to a keyword in the routine to be called, e.g. the HTML code for a pull-down menu for the tree and a field for the shot number would be:

```
Tree: <SELECT NAME="tree" >
          <OPTION VALUE="WF"> WF
          <OPTION VALUE="EFIT"> EFIT
          <OPTION VALUE="ENG"> ENG
      </SELECT> <P>
```

```
Shot: <input TYPE="text" NAME="shot">
```

There must also be an input field in the form (usually hidden) which specifies the name of the IDL routine to be called. When the Submit button is clicked on a Web Tool form, a string like "http://nstxops.pppl.gov:7777/?RoutineToCall=plotip&tree=WF&shot=142000" is sent to the Perl script running on nstxops.pppl.gov and listening on port 7777, in this example (see [12] for the details of Perl, HTML, and IDL code for the boxes shown in Fig. 1). The Perl script processes this command by replacing ampersands in the string with commas and placing quotes around the value for each keyword. The IDL routine, "plotip" in this example, must be in the path defined by IDL_PATH in the context under which the Perl server is running. The called IDL routine must take all its inputs from keywords (or other external sources, like MDSplus, SQL databases, files, or system-wide environmental variables), must not need to use the 'X' device, and must have appropriate defaults for fields where no data was entered. Unspecified fields will usually be a null string, but unchecked boxes will have a value of "off". Most IDL routines used from Web Tools write to conventionally-named log files, though this is not necessary. If the name of the IDL routine contains the string "print" or "list" it is assumed that it will write results to a text file whose name is specified by the keyword "Outfile", and the Perl script will send its contents back to the Web Browser. Otherwise, the Perl script assumes the called IDL routine created a plot in a single graphics window. The contents of that window is made into a Portable Network Graphics (PNG) file which is then sent back to the browser by the Perl script.

The IDL commands sent from the Perl script for plotting operations are to:

1) First plot a generic error message to the uniquely-named PNG filename, in case something goes wrong,
2) Call the specified IDL routine with the key words formed from the inputs on the Web Tool,
3) Write the resulting contents of the graphics window to the named PNG file,
4) Delete the lock file that was created before the call to IDL was made.

The Perl script, which immediately forked a subprocess after the call to IDL, waits for the lock file to be deleted and then sends the PNG file back to the browser.

## 4. Useful features of more advanced Web Tools

The following are some features of the core plotting and listing Web Tools on NSTX. These features do not come automatically for the "easy" interfaces described above, but illustrate how powerful features in existing IDL code have been leveraged for Web Tools, with minimum effort.

- Time scales in math operations are automatically converted
- Publication-quality plots can be emailed as postscript or pdf files
- Signal names and plotting ranges can be pasted from the clipboard
- Previous inputs can be remembered using cookies
- Hot links for plotting are generated from signal-search results
- Data values can be listed in columns for use in other programs such as Excel or KaleidaGraph [13]
- IDL routine names can be used in place of signal names (as in ReviewPlus [14]).
- Only data which can be seen on the screen is sent, for faster plotting of large signals

- Plot summaries are available on Logbook search results
- Tick marks have better spacing and labeling, including better display of exponents
- Shot ranges can be entered with a convenient shorthand notation, and shot numbers can be incremented or decremented by clicking buttons
- Spacing between plot columns and rows can be customized
- JavaScript is used for a variety of functions, such as checking inputs

## 5. Limitations of current Web Tools

When plots are returned to the browser, they are in the form of a PNG file of the requested resolution so they can be transferred and displayed quickly. The disadvantage of this method is that additional detail cannot be displayed without redrawing and resending the image from the server. A prototype was developed where the plotting in the browser was managed by ElVis [15], which stores the data in the applet so the user can pan and zoom with the mouse. This was very slow for large signals, so more work is needed to make this usable.

Web Tools do not update automatically when new data is available, since they are not aware of MDSplus events nor use server-push technology. When a user wants to see new data, they have to refresh the web page (pages could be refreshed on a timer, but this would be inefficient). Although Internet Explorer allows pasting values from the clipboard into fields on the web page via JavaScript, other browsers, like Firefox and Safari, only allow individual fields to be pasted in manually. When a user makes a wrong entry, such as an invalid shot number, Web Tools just indicates that inputs should be checked.  A user would have to look in the log file for the specific problem, which is not trivial.

## 6. Performance

Timing tests were performed in 3 environments, all from applications running on a Linux cluster: 1) over a Local Area Network (100Mbps) to an office PC running X-windows (labeled “LAN X” in Fig. 2), 2) over Wide Area Network to a Verizon fiber optic network to a residence (nominal speed of 25Mbps), and then wirelessly to a PC running X-windows (“Remote X”), and 3) to the same home environment, but using NoMachine NX [16], in which X-traffic is compressed before transmission. All the access to the MDSplus data in these tests is by the “thick client” method, in which all expression evaluation and file I/O is done by the server. Ping times over the LAN are less than 0.4 msec and are around 25 msec over the Remote connection. There is 0% packet loss in all cases.

Users on NSTX typically plot data from MDSplus using the X11-based Dwscope [8] or jScope [16] programs, distributed with MDSplus, or from programs written in IDL. IDL and jScope are slower when plotting large amounts of data remotely because they send all the data. DWSCOPE and Web Tools limit the data sent to screen resolutions.

Web Tools deliver both small and large amounts of data in a reasonable time. As shown in Fig. 2 (note the vertical log scale), plotting MDSplus data from Web Tools is slower than other methods when X traffic is compressed using the NoMachine technology, but faster than some other commonly-used X-windows-based software over a remote connection.

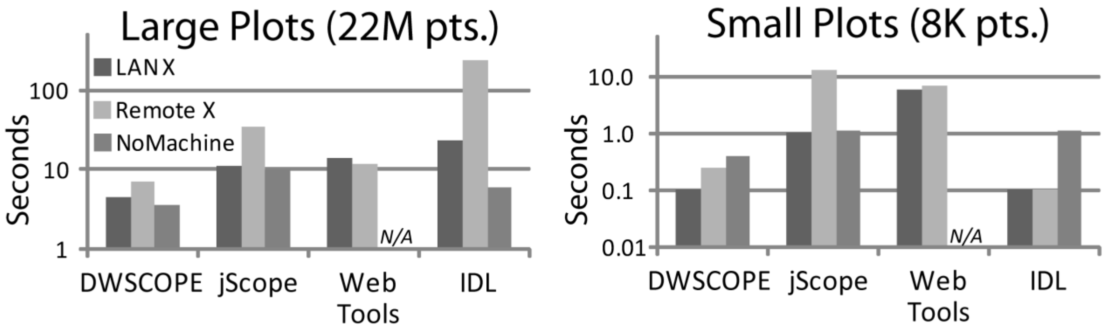

**Fig. 2.** Comparison of times to plot large and small numbers of data samples from an MDSplus tree using Web Tools and 3 X-windows programs. Results are for plots on a LAN (dark), remotely uncompressed (light gray) and remotely using NoMachine (medium gray).

## 7. Evaluation

Web interfaces to existing IDL code can be produced with a minimum of reprogramming effort: less than an hour for a simple interface, to as much as a few days when the IDL code needs to be modified to fit the Web Tools paradigm. Users find the web interfaces intuitive and appreciate not needing usernames or passwords (when inside the firewall), X-servers, or having to remember names of routines. There have been no security issues with Web Tools. From offsite, a user has to authenticate at the PPPL firewall using SecurID. Common security checks are handled in the Perl server, such as preventing buffer overflow, commands to the server's OS, and file-changing commands. Log files show any problems with user entries or code, and lines can be copied and pasted into an interactive IDL session for easy debugging. The signal plotting Web Tools were adapted for remote monitoring of EAST shot data with only a few man-weeks of effort; porting to additional sites should now take even less time. In summary, Web Tools enhance programmer productivity by leveraging existing code and increase user efficiency by providing intuitive user interfaces.

## Acknowledgements

The author thanks the NSTX users of Web Tools for extensive testing and many suggestions over the years. Thanks to Kevin Ying and Tom Carroll from PPPL for adapting Web Tools to work on EAST data. This work was supported by DOE Contract DE-AC02-09-CH11466.

## References

[1] S. Kaye, M. Ono, Y.-K.M.Peng, D.B. Batchelor, M.D. Carter, W. Choe, et al., "The Physics Design of the National Spherical Torus Experiment." *Fusion Technology* **36**, July 1999, p. 16, or http://nstx.pppl.gov/, June, 2011.

[2] IDL (Interactive Data Language)**,** The Data Visualization & Analysis Platform, http://www.ittvis.com/idl/, June, 2001.

[3] J.A. Stillerman, T.W. Fredian, K.A. Klare, G. Manduchi, "MDSplus Data Acquisition System" *Rev. of Sci. Instrum.* **68** (1) January 1997, p. 939, or http://www.mdsplus.org/, June, 2011.

[4] W. Davis, P. Roney, T. Carroll, T. Gibney, D. Mastrovito, "The use of MDSplus on NSTX at PPPL" *Fusion Eng. Des*. **60** (2002), 247-251.

[5] EPICS (Experimental Physics and Industrial Control System), http://www.aps.anl.gov/epics/, June, 2011.

[6] P. Sichta, W.M. Davis, J. Dong, D. Mastrovito, G. Tchilinguirian, G. Zimmer, "A Decade of NSTX Operations Using Collaborative Software" 23rd Symposium on Fusion Engineering, San Diego, CA (2009)

[7] NSTX Web Tools: http://nstx.pppl.gov/nstx/Software/WebTools/ June, 2011.

[8] ***Scope*** an MDSplus Plotting Utility: http://www.mdsplus.org/index.php/Documentation:Beginners:Scope, January, 2012.

[9] S.A. Sabbagh, S.M. Kaye, J. Menard, et al., Equilibrium Properties of Spherical Torus Plasmas in NSTX, *Nucl. Fusion* **41** (2001) 1601.

[10] Homepage for the LRDFit code: http://w3.pppl.gov/~jmenard/software/lrdfit/lrdfit-index.htm, January, 2012.

[11] TRANSP Home Page: http://w3.pppl.gov/transp/, January, 2012.

[12] http://nstx.pppl.gov/nstx/Software/pdf_files/WebToolsCodeDataFlow.pdf, July, 2011.

[13] KaleidaGraph: http://www.synergy.com/, January, 2012.

[14] J. Schachter, "Data Analysis Software Tools for Enhanced Collaboration at the DIII–D National Fusion Facility," *Fusion Eng. Des.* **48** (2000), 91-98.

[15] ElVis, Scientific Graphics for Visualization and Monitoring: http://w3.pppl.gov/elvis, by Eliot Feibush. June, 2011.

[16] NoMachine NX: Desktop Virtualization and Remote Access Management Software, http://www.nomachine.com/, March, 2012.

[17] jScope: http://www.mdsplus.org/index.php/Documentation:Beginners:JScope, January, 2012.