ELSEVIER

The World Wide Web as a graphical user interface to program macros for molecular graphics, molecular modeling, and structure-based drug design

Neil R. Taylor and Ryan Smith

Biomolecular Structure, Glaxo Wellcome Research and Development, Medicines Research Centre, Stevenage SG1 2NY, Hertfordshire, UK

In this article we describe how the World Wide Web (WWW or Web) has been employed to provide access to computational chemistry software and protein structure data via program macros. We show how the combination of Web technology and macros can automate both the running of chemistry software and the execution of complex operations on protein structures. The current version of the system supports the molecular visualization packages GRASP,[1] RASMOL,[2] MOLVIEWER-OGL[3] and INSIGHT95,[4] and the ligand design tool GRID[5] and includes more than 175 in-house protein–ligand complexes. The approach enables inexperienced users to confidently make full use of sophisticated modeling techniques by offering only sensible options, hiding parameter settings, and controlling program invocation and macro execution. Our interface provides both the expert and non–expert alike with powerful tools for protein structure visualization, molecular modeling, and rational drug design. © 1996 by Elsevier Science Inc.

Keywords: molecular graphics, World Wide Web, macros, graphical user interface

INTRODUCTION

Historically, the use of molecular graphics, molecular modeling, and chemical information software has been limited to the experienced user who is familiar with the background concepts, specific user interfaces, and the overall computing environment. Furthermore, the use of most software packages has been labor intensive, with considerable time and effort often being required for computer-based approaches to molecular problems. One solution to the second problem has been the use of program macros (also known as *scripts*). These are programs consisting of commands for running and controlling software packages. Our aim has been to address both issues by providing macros as end-user tools so that users at any level may safely and correctly apply the most advanced features of sophisticated modeling software.

The incredible rise in popularity of the World Wide Web has resulted in a revolution in the way computers are used and the way information is accessed. An important part of the success of the Web is the Common Gateway Interface (CGI)[6] which allows a Web browser to be used as an interface to software packages. The ease of use of the Web and the global familiarity with browsers make it an attractive environment for delivering software tools to a wide audience. This article details the methods we employed for distributing program macros to the wider scientific community using the World Wide Web. These methods are an extension of some of the ideas previously reported by Casher et al. in their implementation of hyperactive molecules.[7]

The modeling we describe herein focuses on the interpretation of protein structure information and its use in receptor-based drug design. We have sought to obtain maximum benefit from specialized protein structure visualization software, including GRASP,[1] which is particularly suited to mapping chemical properties onto molecular surfaces; RASMOL,[2] a widely used package for protein structure

Color Plates for this article are on pages 280–282.

Address reprint requests to: Dr. Taylor, Biomolecular Structure, Glaxo Wellcome Research and Development, Medicines Research Centre, Gunnels Wood Road, Stevenage SG1 2NY, Hertfordshire, UK.

Received 29 July 1996; accepted 30 October 1996.

655 Avenue of the Americas, New York, NY 10010

0263–7855/97/$15.00
PII S0263–7855(96)00077-X

viewing; MOLVIEWER-OGL,[3] which we have found to be useful for highlighting individual residues in protein–ligand complexes; and INSIGHT95,[4] a comprehensive molecular modeling package that is used for the superimposition of multiple structures. In each case, the commands and parameters associated with the running of the programs are hidden from the end user. The current version of the interface similarly provides easy access to the ligand design tool GRID.[5] We have attempted to create a system that offers the user a simple, intuitive set of options that rapidly delivers meaningful results. In addition, these results are presented within a complete running version of the software, thereby allowing further interactivity.

METHODS

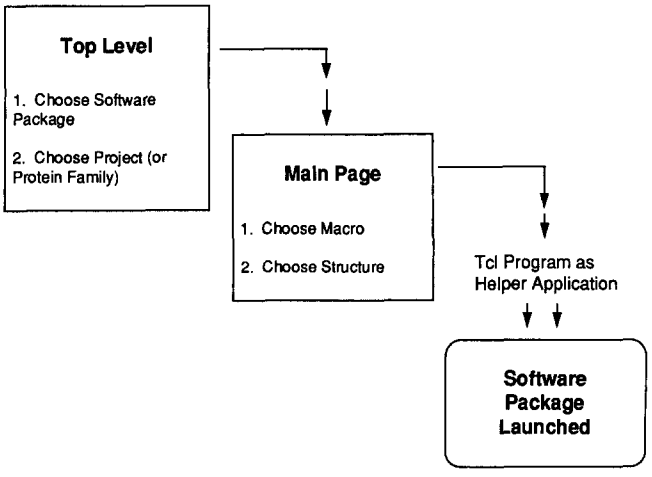
Overview

At the core of our approach is a set of macros that perform specific operations on protein structures. The power of macros is that they enable complex operations that normally require many steps to be performed with single commands. Each macro is written in a command language that is specific to each modeling package. In some cases a macro may be only two commands, for instance, highlighting an amino acid residue in the active site of a protein, while in other cases it may consist of hundreds of commands, for example, reading a protein–ligand complex into a graphics package and displaying the molecular surface of the protein in the vicinity of the active site, color coded according to electrostatic potential.

A Web browser can be used as a graphical user interface (GUI) via the Common Gateway Interface and controlled by a scripting language (we use Tcl, Tool Command Language[8]). An HTML[6] document that is used to pass parameters to a CGI program is known as a *form.* Parameters selected within our forms are processed by Tcl programs and (1) create subsequent Web pages (which are usually forms themselves), (2) extract protein coordinates and program macros from local directories and write data to temporary files on local, cross-mounted disk space, (3) create text files required by the graphics and modeling packages, and (4) launch the packages. All temporary files and directories created for a session are deleted by the same Tcl program that launches the packages (see below).

Organization

Scheme 1 shows a summary of the steps that provide access to the program macros and to the display of protein structures. The Top Level Web page, which is illustrated in Color Plate 1, is a form that contains two pull-down menus, one for selecting a molecular graphics program (for example, GRASP, INSIGHT95, MOLVIEWER-OGL, or RASMOL) and one for the medicinal chemistry project (or protein family) from which the structure is to be visualized. A CGI program processes this form and creates a new Web form, the Main Page, which displays the range of macros associated with the selected package and lists the structures associated with the project.



Scheme 1

A sample Main Page, for the program GRASP,[1] is given in Color Plate 2. The macros available are illustrated by pictorial examples. (These are hyperlinked to documents containing the macro source code.) A radio button is linked to each macro and the different protein structures are listed in a scroll-down menu. By default, only the bound ligand and active site of the protein are displayed, although the user can view the entire protein if required. When this form is submitted the software package is launched, the selected structure is read from a local database, and the macro is executed. The selected viewing mode is created "on the fly" and the complete running version of the package is presented on the desktop. Color Plate 3 shows the result of selecting the macro for viewing the molecular surface around the active site color coded by electrostatic potential. At the bottom of the graphics window a short description of display mode is printed—a feature that is also controlled by the macro.

Process control

All of the modeling packages are executed by a single Tcl program, *Wlaunch,* which is defined as a helper application in the *.mailcap* file, and using a server-defined MIME (Multipurpose Internet Mail Extensions[9]) type. Web servers use MIME to inform Web browsers how to deal with different file types. In Unix, a file requested by a browser is cached in a local temporary directory, and then the browser scans the *.mailcap* file for the application that supports the file type. The browser launches the selected application and sends to it the file from the local temporary directory. Our Tcl helper application *Wlaunch* receives text files that have the string .wlaunch at the end of the file name. The information in these files tells the Tcl program *Wlaunch* what software package to invoke, the machine on which to run the package, and where to find the required data. Scheme 2 lists the main part of the program *Wlaunch.*

The procedures that launch the modeling packages consist mainly of commands for reading data from the same file as the main part of the program and one command for launching the software. Scheme 3 lists the procedure that

```
# Tcl helper application Wlaunch for controlling
# the launching of modelling packages
#
# MAIN PROGRAM START

     set inputFile [lindex $argv 0]
     set inFile [open $inputFile "r"]
     gets $inFile programName
     close $inFile

     switch $programName {
          {grasp}        {run_Grasp $inputFile}
          {molviewer}    {run_Molviewer $inputFile}
          {insight95}    {run_Insight95 $inputFile}
          {rasmol}       {run_Rasmol $inputFile}
          }

# MAIN PROGRAM END
```

Scheme 2

controls the running of GRASP. This procedure opens a window on the desktop and executes a short shell (csh) script; in this example *rungrasp.csh,* listed in Scheme 4. The shell script command invokes the software, which then reads the protein structure and executes the program macro.

We use interprocess control (IPC) so that the Web browser can control graphics packages that have already been invoked. In these instances, commands that change the state of an application are sent across the network in the form of X *events* (see Ref. 10 for additional information on

```
# Tcl procedure within Wlaunch for controlling
# the launching of GRASP
#
# RUN_GRASP START

proc run_Grasp {inputFile}  {

    set infile [open $inputFile "r"]
    gets $infile null
    gets $infile tempDirectory
    gets $infile pdbName
    gets $infile macroName
    gets $infile hostName
    close $infile

    exec xwsh -e /public/grasp/rungrasp.csh \
          $tempDirectory $pdbName $macroName $hostName

# delete temporary files when GRASP exits

    cd ..
    exec rm -rf $tempDirectory

}

# RUN_GRASP END
```

Scheme 3

```
# Shell script rungrasp.csh for launching GRASP
#
# RUNGRASP.CSH START

setenv HOST $4
setenv GRASP /public/grasp/data/
cd $1
exec /public/grasp/GRASP -rx $1/$3 $1/$2

# RUNGRASP.CSH END
```

Scheme 4

the X Window system). A similar system has been described previously for hyperlinking a nuclear magnetic resonance (NMR) spectrum with a graphics display process.[7] We currently use two programs for sending X *events: Xsend,* written by R. Sayle (Glaxo-Wellcome R&D, Stevenage Hertfordshire, UK); and *Tksend,* written by M. Hartshorn (Department of Chemistry, University of York, Heslington, York, UK). These programs are controlled in the same manner as the software packages themselves—by using procedures within *Wlaunch.* A CGI program creates a short text file that tells *Wlaunch* which program to use to send the X *event;* the window (modeling package) on the desktop to which to send the command; and the command itself. Examples of where we apply IPC include the software packages MOLVIEWER-OGL and INSIGHT95 (see below).

Protein data

To access crystal structures rapidly, all files containing protein atom coordinates are stored on a local network. Furthermore, the in-house protein structural data is preprocessed—in the *.pdb* files we explicitly define ligands in active sites with HETATM records and all other atoms with ATOM records. This enables the active sites of proteins to be identified immediately. Extra information about the structures, necessary for certain program macros, is coded into the CGI programs. This information includes tables of conserved active site residues (used for the superimposition of related proteins), the names of structures that do not contain waters, and the names of dimeric proteins.

APPLICATIONS

Rational drug design

The tools described in this article were created specifically to aid in the interpretation of protein structure data and receptor-based drug design. A typical receptor-based project is an interactive procedure, in which the steps of protein–ligand structure determination, molecular modeling, synthesis, and biological evaluation are repeated until the binding affinity of the lead compound has been optimized. The role of the computational chemist in this cycle is to use molecular graphics and ligand design software to aid in the interpretation of structural information and to identify suitable modifications for increasing binding affinity. Other

members of the research team, in particular the synthetic medicinal chemists, are often unable to contribute fully their expertise to the ligand design phase because the software packages are highly sophisticated and require a high level of end-user expertise. Our aim has been to reduce the expertise required to access the structural information and advanced software features, thereby enabling all members of a research team to contribute to receptor-based drug discovery.

Visualization

GRASP Color Plate 2 illustrates the Web page associated with the program GRASP (by A. Nicholls et al.[1]). GRASP provides fast and powerful surface rendering for proteins. Our default macro displays only the active site amino acid residues and water molecules, color coded by atom type, and the bound ligand, with its atoms highlighted. Three of our GRASP macros map atomic properties to the molecular surface. These properties are temperature factor, electrostatic potential (calculated using the Poisson equation), and hydrophobicity. The electrostatic properties of proteins can also be visualized as contours of equipotental. An extremely useful macro for drug design enables cavities adjacent to the bound ligand to be visualized. These cavities show where functional groups on the ligand might be extended to give improved contacts with the protein, or where bridging water molecules might reside. The other macros we have written enable color coding of the protein molecular surface according to distances between the van der Waals surfaces of the ligand and protein atoms, and color coding of the surface according to curvature.

MOLVIEWER-OGL MOLVIEWER-OGL, by M. Hartshorn,[3] is a molecular graphics package for proteins and DNA, created for virtual reality applications, that has proven to be a useful tool for looking at structures. The package has an easy-to-use Toolkit (Tk) interface, a Tcl-based scripting language, and the ability to communicate with Web browsers; all of which enables us to make full use of the functionality embedded in the software using our Web interface. We have combined MOLVIEWER-OGL with a Tcl script that "marks up" the contents of .pdb files and enables individual residues, and groups of residues, to be highlighted in the graphics window from a Web browser. Color Plate 4 shows a sample Web form that controls MOLVIEWER-OGL, and the graphics window is illustrated in Color Plate 5. The protein displayed is a complex containing influenza virus neuraminidase.[11,12] When the cursor is placed over one of the single-letter symbols in the browser, the three-letter symbol and number of the residue are displayed as part of the Uniform Resource Locator (URL)[6] (usually listed at the bottom of the Web browser; see Color Plate 4). Clicking the mouse highlights the selected residue in the graphics window. This feature facilitates the identification of specific residues in proteins that are unfamiliar to the user. Interprocess communication between the Web browser and the graphics application is controlled by the program *Tksend* (also by M. Hartshorn).

MOLVIEWER-OGL includes powerful surface display capabilities, and the Web GUI enables the molecular surface around the active site to be displayed and colored according to electrostatic potential. Another feature of MOLVIEWER-OGL and the Web GUI is the ability to display protein secondary structure. We have extended the capabilities of our Tcl scripts that control MOLVIEWER-OGL from the Web and enable any structure from a locally installed copy of the Brookhaven Data Bank[13] to be manipulated using the same format as shown in Color Plate 4.

INSIGHT95 INSIGHT95, by Molecular Simulations, Inc. (MSI),[4] is currently included in our Web GUI. The major INSIGHT95 macro displays a bound ligand with carbon atoms highlighted and the surrounding active site residues color coded by atom type. A powerful feature that we have included in the INSIGHT95 Web GUI is the ability to superimpose different protein-ligand complexes onto one another. Once a structure is displayed in INSIGHT95, the Web interface can be used to select a second complex to overlay onto the first (thereafter, any number of complexes can be overlaid). Communication from the Web browser to the INSIGHT95 command line is controlled by our CGI programs and facilitated by the program *Xsend,* by R. Sayle. Superimpositions are based on lists of conserved active site residues. The automatic superimposition of protein structures saves significant amounts of time and work, particularly when related proteins have different residue labeling schemes, or contain mutations, insertions, or deletions. The approach also facilitates the docking of ligands across structurally related proteins.

RASMOL RASMOL, by Sayle and Milner-White,[2] is a widely used program for protein visualization, and the scripting capabilities have enabled us to add this package to the Web interface. We provide a GUI similar to that for GRASP, with a pictorial menu of viewing modes, for displaying either active sites or entire proteins, colored by various residue-based, chemical properties. Coloring schemes include atom type, temperature factor, and hydrophobicity.

Molecular modeling

GRID We have designed a simple Web GUI for the ligand design tool GRID (Goodford[5]). Only two selections are required from the Web to run GRID: (1) the protein structure and (2) the probe sphere. The CGI Tcl programs create all of the input files required by GRID; execute the GRID calculation; launch a graphics package (in our case INSIGHT95, although any one of a number of packages could be used); and execute a program macro for reading in the protein coordinates and GRID map, and displaying the active site and GRID contour. The results from GRID are automatically contoured for inspection at a default energy level (specific to the probe sphere selected) and then a sliding ruler widget is provided for recalculating contours at different energy levels (the ruler is specific to INSIGHT95). The sliding ruler is created automatically at each session and is scaled according to the results from the GRID calculation. The most favorable region of binding for a probe sphere can be viewed by sliding the ruler to the low-energy

end of the scale (most negative value), and by sliding the ruler to the other end of the scale, the molecular surface (or "probe-accessible" surface) can be displayed.

ADVANTAGES AND DISADVANTAGES

There are many advantages to using the Web as a GUI to program macros and computational chemistry software. These advantages include the following: (1) the process of viewing structures and their properties is fast; (2) access to the protein data and software tools is opened to a wider audience; (3) data are managed efficiently owing to reduced duplication of files; (4) the time required to gain experience with software is reduced, thereby lowering training costs; (5) many changes associated with running modified/upgraded software packages are hidden from the inexperienced end user; (6) the implementation of state-of-the-art techniques is usually straightforward: and (7) documentation and manual pages can be easily embedded into the GUI in the form of hyperlinks. Furthermore, one important advantage of using a Tcl program as a helper application is that we are able to add new packages to the GUI, and modify the running of existing packages, without the need to continually update the *.mailcap* file. (The *.mailcap* file requires modification only once—the first time the Web GUI is used.)

There are, of course, some disadvantages in our methods, such as machine dependency. That is, the software packages we have implemented run only on Silicon Graphics workstations. In general this is not a serious issue, as workstations are the standard for molecular modeling (lower specification machines are often unable to deliver the graphics performance required for advanced protein structure visualization). We are considering removing the platform dependency specifically for the presentation of results through the use of VRML (Virtual Reality Modeling Language).[14] Another limitation is that the techniques reported are suited to local intranet systems, not the whole World Wide Web. In principle, however, most of the methods we have described could be extended to the Internet. This may be done, for example, by enabling remote users to download setup files so that their systems can be configured in an identical manner. Despite the minor drawbacks, the tremendous gains described above (advantages 1–7) ensure the future success of using the Web as an interface to molecular modeling software.

CONCLUSIONS

World Wide Web technology has been used to distribute sophisticated tools for molecular graphics, molecular modeling, and rational drug design. Complex program macros for controlling software packages, constructed for visualizing protein structures, can be selected in a simple and intuitive manner from the Web. CGI programs combine selected structures with selected viewing modes and launch the software packages on the desktop. This approach has been successful for delivering complex structural information to a wide audience in a format that enables novice users to interrogate the data. We intend to extend the current system to include tools for two- and three-dimensional database searching, flexible docking, and combinatorial library design.

At present there is tremendous interest in the Web and the technology is evolving rapidly. During the time of preparing this manuscript the object-oriented programming language Java[15] appeared and it has already excited great interest. Java enhances the interactivity of Web documents and the way the Web is used to provide access to software packages. Significant developments are also occurring in the area of distributed objects, and the application of object communication facilities, as defined by the Object Management Group (OMG) Common Object Request Broker Architecture (CORBA) specification,[16] will further enhance the utility of Web-based software. Finally, within the chemistry domain, the Chemical Exchange (CEX) protocol offers interoperability between computational chemistry packages across the Web.[17] It is becoming increasingly apparent that using the Web as an interface to computational chemistry software will be of key importance in the future of chemical information technology.

ACKNOWLEDGMENTS

The authors thank Roger Sayle for his program *Xsend*, Anthony Nicholls for help with several of the program macros for GRASP, Mike Hartshorn for providing us with the programs MOLVIEWER-OGL and *Tksend*, and Stuart Moodie for many useful discussions and assistance with the manuscript.

REFERENCES

1 Nicholls, A. Department of Biochemistry and Molecular Biophysics, Columbia University, New York 10032 (nicholls@cuhhca.hhmi.columbia)
2 Sayle, R.A., and Milner-White E.J. RASMOL: Biomolecular graphics for all. *Trends Biochem. Sci.* 1995, **20,** 374–376
3 Hartshorn, M.J. Department of Chemistry, University of York, Heslington, York, UK (mjh@yorvic.york.ac.uk)
4 Molecular Simulations, Inc. *Insight II 95.0 User Guide.* Molecular Simulations, Inc., San Diego, CA 1995
5 Goodford, P. A computational procedure for determining energetically favourable binding sites on biologically important macromolecules. *J. Med. Chem.* 1985, **28,** 849–857
6 For information on the World Wide Web, HTML, and CGI see Liu, C., Peek, J., Jones, R., Buus, B., and Nye, B. *Managing Internet Information Services.* O'Reilly and Associates, Sabastapol, CA, 1994
7 Casher, O., Chandramohan, G.K., Hargraves, M.J., Leach, C., Murray-Rust, P., Rzepa, H.S., Sayle, R., and Whitaker, B.J. Hyperactive molecules and the World-Wide-Web information system. *J. Chem. Soc. Perkin Trans.* 1995, **2,** 7–11
8 Ousterhout, J.K. *Tcl and the Tk Toolkit.* Addison-Wesley, Reading, Massachusetts, 1994.
9 Borenstein, N., and Freed, N. MIME (Multipurpose Internet Mail Extensions). 1. Mechanism for specifying and describing the format of Internet message bodies. (Internet Request for Comment, No. 1521) 1993

10 Scheifler, R.W., and Gettys, J. *The X Window System,* 3rd Ed. Digital Press, Newton, MA, 1992
11 Varghese, J.N., Laver, W.G., and Colman, P.M. Structure of the influenza virus glycoprotein antigen neuraminidase at 2.9 Å resolution. *Nature (London)* 1983, **303,** 35–40
12 Colman, P.M., Varghese, J.N., and Laver, W.G. Structure of the catalytic and antigenic sites in influenza virus neuraminidase. *Nature (London)* 1983, **303,** 41–44
13 Bernstein, F.C., Koetzal, T.F., Williams, G.J.B., Meyer, E.F., Brice, M.D., Rodgers, J.R., Kennard, O., Shimanouchi, K., and Tasumi, M.J. The Protein Data Bank: A computer-based archival file for macromolecular structures. *J. Mol. Biol.* 1977, **112,** 535–542
14 Vollhardt, H., Henn, C., Moeckel, G., Teschner, M., and Brickmann, J. Virtual Reality Modelling Language in chemistry. *J Mol. Graphics* 1995, **13,** 368–372
15 See Java Home Page: http://java.sun.com/
16 See Object Management Group Home Page: http//www.omg.org/
17 See Chemical Exchange Home Page: http://cgl.ucsf.edu/cex/