A Computer-Aided Drug Discovery System for Chemistry Teaching

Robert Gledhill,[†] Sarah Kent,[†] Brian Hudson,[†] W. Graham Richards,[‡] Jonathan W. Essex,[†] and Jeremy G. Frey*[,†]

School of Chemistry, University of Southampton, Southampton SO17 1BJ, United Kingdom, and Central Chemistry Laboratory, South Parks Road, Oxford OX1 3QH, United Kingdom

Received September 8, 2005

The *Schools Malaria Project* (http://emalaria.soton.ac.uk/) brings together school students with university researchers in the hunt for a new antimalaria drug. The design challenge being offered to students is to use a distributed drug search and selection system to design potential antimalaria drugs. The system is accessed via a Web interface. This e-science project displays the results of the trials in an accessible manner, giving students an opportunity for discussion and debate both with peers and with the university contacts. The project has been implemented by using distributed computing techniques, spreading computer load over a network of machines that cross institutional boundaries, forming a grid. This provides access to greater computing power and allows a much more complex and detailed formulation of the drug design problem to be tackled for research, teaching, and learning.

INTRODUCTION

Recent developments in methodology have given rise to a range of approaches to research referred to in the U. K. as “e-science”. These typically involve heavy use of computational and network resources and depend on complex infrastructure mediated by transparent middleware. This can be viewed as the science that runs over the infrastructure referred to as the Grid. In the U. S. A., a similar concept is described as the “Cyber-infrastructure”. The e-Malaria project[1] uses these e-science techniques and technology to make modern chemical ideas and practice available to a much wider audience of less-experienced scientific investigators: students. It also serves as a device to connect these students with active researchers—a virtual organization for scientific research, teaching, and learning.

Computer technology is used to an increasing extent in the field of drug research. Typical applications include databases to hold information about the known properties of molecules, 3D visualization systems to aid understanding of protein binding sites and their interactions, and drug docking calculations; see, for example, Tai et al.[2] Despite these developments, though, the possibilities of using these modern tools for teaching chemistry, and particularly drug design, have been little explored, even though Internet-based distance-learning courses have already been introduced.[3] Also, and just as importantly, the project serves as a test bed for the development of new Web-based tools and interfaces for the software tools used in pharmaceutical research. Leveraging many existing technologies in creating this site, we have gained experience integrating chemical software in a distributed computing environment to create a research tool with a seamless and straightforward user interface. The knowledge gained from doing this should help with further systems integration work—the construction of more complex, database-enabled chemoinformatic tools.

Other projects have used distributed computing techniques to solve large, coarsely parallelizable problems. Generally, these projects take the form of a “screensaver”, by which participants can donate unused time on their computers for problem solving. The first of these to draw wide attention was the SETI@home project,[4] whose screensaver program is designed to process signals received from radio telescopes, searching them for signs of intelligent life. More recently, a number of large-scale scientific projects based on either the United Devices (UD) platform (see below) or the Berkeley Open Infrastructure for Network Computing (BOINC)[5] have been created. These include einstein@home[6] (searching for pulsar gravitational waves in data from Michelson interferometers distributed around the world), ClimatePrediction.net[7] (creating a large ensemble of long-period climate simulations to aid model parametrization and validation), and the “Screensaver-Lifesaver” project[8] run by the University of Oxford. This latter project aims to investigate potential drugs for cancer and various other diseases, notably anthrax. The distribution of the docking code is managed by the United Devices platform, and the docking itself is performed by the THINK and LIGANDFIT codes. This project is concerned with testing very large numbers of compounds, with the goal of discovering genuine drug candidates, and to this end, there is relatively little interactivity for users. More recently, a molecule editor program has been added to the main Web site, so users may sketch molecules and have them docked against the target proteins, but the purpose of the project is primarily drug discovery, not teaching. Inspired by this, we have developed an e-learning strand to extend this line of research.

e-Malaria is the first project its authors are aware of which seeks to combine multiple pieces of professional pharmaceutical research software into a framework for instruction. While some previous projects such as SETI@home and

* Corresponding author. Tel: +44 (0) 2380 593209. E-mail: J.G.Frey@soton.ac.uk.
[†] University of Southampton.
[‡] Central Chemistry Laboratory.

Figure 1. Female *Anopheles gambiae* mosquito—a malaria vector. Image courtesy of Christopher Curtis at the London School of Hygiene and Tropical Medicine.

Graham Richards's molecular docking project have used a similar approach to distributed computation, these have not been aimed primarily at chemistry teaching.

A software environment integrating a Web interface, a database, and a distributed computing infrastructure has been developed. The software is aimed at high school (16−18 years old) students of chemistry; the students are asked to design chemical compounds using a sketchpad that they can then submit for docking against a known malaria target. The score from their docked structure may then be used together with molecular graphics to further refine their potential drug. This software teaches the elements of molecular structure and intermolecular forces, with the added driver of targeting a serious illness. (Figure 1 shows a common malaria vector.)

Enabling dozens of people to tackle problems in rational drug design requires a large amount of computer power, as each docking job takes, on average, 5 min. This is more than could be supplied directly by the designers of the project and becomes more severe as the project scales up toward regional and national deployment. This problem was solved by distributing the computing tasks across machines both within and outside our immediate organization, at the expense of increasing the level of complexity of the system to enable simple administration and to address security concerns. By choosing this path, we greatly extended the detail and depth of the investigations that can be performed but, at the same time, created new organizational challenges. To maintain the project, we need to maintain the support of the organizations that allow us to distribute the workload across their machines. This virtual organization and the consequent administrative boundaries is one the features that makes e-science such an interesting challenge for project organizers. It also imposes constraints on the designers of the software system, as will be explained below.

Many organizations around the world are attempting to address the malaria problem. Some of these are charities, some are backed by government funding, and others have corporate sponsors. The principal, coordinating agency is the international World Health Organization. Another large charitable body is the Malaria Foundation International, which is funded by numerous pharmaceutical companies and government bodies. Also important is the Medicines for Malaria Venture, a charitable foundation which "finds its origins in the failure of the market system to provide the required incentives for wide scale R&D in new medicines to treat malaria" and wishes to bring together public and private sector researchers to address this. All of the above organizations are linked to the e-Malaria home page, along with the Wellcome Trust's excellent Web resource on the disease and their research into it.

This paper will now describe the following four stages of the e-Malaria project, rounding off with some of the details of the hardware setup and software processes:

(i) Planning and specification.

(ii) Development of a prototype.

(iii) Testing and appraisal.

(iv) Development of the completed system.

PLANNING AND SPECIFICATION

The system we were interested in creating would enable students to design small molecules of some description and dock them against the malarial dihydrofolate reductase (DHFR) enzyme, whose structure has recently been solved.[9] Also, alongside this core activity would be a collection of teaching material about malaria tailored for the curriculum of the target students. This teaching material would ideally include some further interactive material, such as quizzes.

On initial consideration, we decided that creating a system for docking tri- and tetrapeptides would be ideal for two reasons. First, it would be straightforward for such molecules to be made if any of the students' docking studies showed promise, and this obviously offers a strong encouragement to the participants. Second, it was perceived that the creation and docking of arbitrary molecules would be a particularly hard task to achieve, not least because of the difficulties of creating a Web-based interface for designing them. A small peptide-based system would actually allow us to construct molecule files for all possible molecules in advance of the

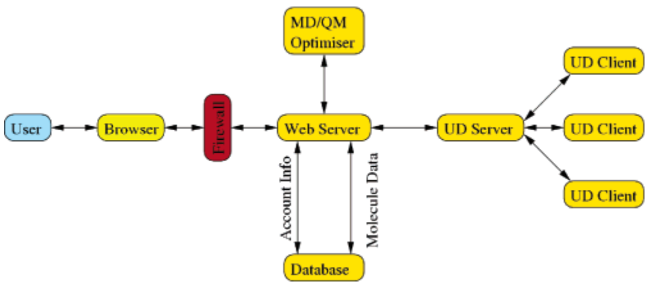


Figure 2. High-level schematic of the e-Malaria system illustrating the flow of information in the system.

more computationally intensive docking and to store them in a database.

An important consideration in the design and layout of the site was that of accessibility for children with special learning needs, particularly dyslexia. To arrange this, it was envisaged that cascading style sheets (CSSs) would be used to create a distinct visual design to aid such students.

To speed development and reduce costs, it was decided to put the system together using as much pre-existing software as possible, with a strong preference for software with open source or free licenses. Where custom software was necessary, it would be written, but this would be kept to a minimum. As demands on the hardware increase, the server can be upgraded; however, it was obvious from the start that much of the calculation needed would have to be distributed across machines outside of our immediate organization.

DEVELOPMENT OF A PROTOTYPE

From these requirements, the general outline of a system became clear, as illustrated in Figure 2; a number of further considerations were also apparent.

The site is built using a number of database-intensive common gateway interface (CGI) scripts; the speed and nature of the Web server employed is a crucial factor in the usability of the system. The Apache Web server is used to host the site, because of its good track record and open source nature. To maximize the performance of the scripts running the site, the "fast-CGI" Apache module and Perl interface were employed to reduce the overheads in restarting server scripts requiring database connections. In the database arena, there are two serious, large-scale offerings that are open source: MySQL and Postgres. From these, we chose MySQL, principally on the grounds that it was being used in another project and we save administration costs by running both on the same database.

To maximize the educational value of the ligand design and docking exercise, it was highly desirable to have some means of viewing the structures of molecules in a three-dimensional format. For ordinary chemical research work, there are many such viewing programs available, such as VMD, Pymol, and so forth. For a strictly Web-based interface, there were only two serious alternatives: MDL's Chime and OpenScience's Jmol.[10] MDL Chime, while being quite complete and having the benefit of years of testing, has a number of issues preventing our using it. First, although free for noncommercial use, it labors under a non-open-source license that restricts our methods of distributing it to third parties. Also, the free version of Chime cannot be used to retrieve files from a database: a separate product called Chime Server must be purchased for this functionality (Chime is not currently available for the Linux or Macintosh operating systems). Jmol is an open source application which meets most of our requirements with the only missing features that would have been useful at the time of implementing the software being specifiable hydrogen bonds, molecular surfaces, and orbital surfaces. Jmol has since proven to be a robust and high-quality piece of software. As Jmol is based on the Java platform, it is relatively easy to distribute and is platform-independent

Adopting the Java platform provides access to the ACD/labs' "structure drawing applet",[11] which allows the user to draw arbitrary molecules and output a Sybyl "mol" file describing them. It is simple to use for creating Web pages and controllable from Javascript, allowing more complicated Web page applications to be built. The only comparable alternative editor package found was JchemPaint,[12] from the same OpenScience project as Jmol, which may be incorporated in the future. "MarvinSketch"[13] from ChemAxon is developing rapidly and provides some addition functionality for names and shows that this is an active area of development which needs to be kept under review. The 2D sketch needs to be converted to a 3D molecule with correct geometry. Subsequent developments of chemical graphical input programs are beginning to incorporate this conversion, but at the time, these were not viable, and the approach we took to this extra step is described in detail below.

During the development of the prototype system, the university's dyslexic teaching center was consulted about the best practice for dyslexic-friendly Web site layout and typography. A long list of guidelines was obtained, and these guidelines were adhered to as closely as possible in the development of a CSS style sheet for students with the condition. Obviously, this created a further requirement for the Web site to use CSS as consistently as possible.

TESTING AND APPRAISAL

Once the prototype system had been completed, a number of external reviewers were contacted to test the site, comment on its usability and stability, and help find any problems with the software components or system design. Their feedback was extremely helpful in improving the system, finding software errors, and optimizing the user interface.

Two sessions were undertaken at Havant College with students who had just completed the U. K. AS-level chemistry syllabus. The students were of mixed ability, and there were approximately 20 students in each session.

The sessions were intended to supplement the students' study of the way medicines are designed and fitted into a series of talks given by industry representatives.

The sessions consisted of a talk lasting 45 min on the background of the project and how computational drug design is used within industry. This was followed by a practical session in which students got to grips with the system and used a current drug as the start point for an investigation into the structural features that form a pharmacophore for antimalarial drugs.

The system was instantly understood by many of the students and was easily explained to those who were less confident with computer programs. Most students spent some time using the system to get accustomed to its controls. The system was well-received by the students, who were pleased to get some experience in authentic tasks, although some were unnerved that there was no answer expected for the task set and that, while the instructor could offer advice, they could not give a definitive answer. It was suggested that work on the system could be used toward key skills evidence in IT that can be used to contribute points toward university entry, as well as provide a useful tool for illustrating many structural topics in chemistry such as the 3D shapes of molecules, chirality, and structural groups. The teacher believed the system could be used to aid the presentation of reports and could aid revision for students without access to molecular modeling kits. In summary, the system worked well, was easy to understand, and offered the students a new tool for chemical understanding. It is hoped that the system tools will be useful in many projects besides its intended use.

DEVELOPMENT OF THE COMPLETED SYSTEM

Administration. Extensibility of the system to many users, sites, and computational resources had become an essential prerequisite of the design. To manage this, a sophisticated and secure administration system was required. The users of the system were classified in terms of student "designer" accounts, "teacher"-level administrator accounts, "site"-level administrator accounts (envisioned as a senior teacher who would provide a single point of contact at each site), and a "root" account from which top-level administration tasks would be performed. At this stage in the design process, we were faced with the twin considerations of Data Protection Act compliance and the responsibilities of storing any type of information about minors on a computer system. To tackle the first of these, we took the straightforward approach of providing a record of the information stored about any user account on the system on their "logout" page. To address the second, we decided to store the absolute minimum of information about any user in the system: a login name, a password, their login expiry date, and the molecule data they have created. Teachers are responsible for creating accounts for their students and are free to use any type of name or pseudonym for their charges.

Shift to Client-Side Rendering. When first creating the site, it was decided, without much critical thought, to construct the pages using very similar techniques to those of most other Web sites. The page headers and footers containing information about the user would be constructed on the fly by a CGI script, while the sidebar and document content would be added by the server itself, as directed by Server Side Include directives issued by the CGI. Complex content, requiring information from the database, would be constructed by CGI modules included from the master script.

Following this approach, the navigation bar content for each page of the site, as seen by each user type, was originally generated as a large set of partial HTML files to be included by the server. A script was written to build these files automatically and place them into a directory structure calculable by the CGI. It was found after a while that this constructor script required considerable maintenance and was prone to error. To fix this, a client-side constructor script was put into place, containing all the data on user permissions and page titles to construct a navigation sidebar at page view time. This had several interesting features:

(i) The script did not change between pages and, so, could be cached on the remote browser, reducing load on the server.

(ii) It provided a single point of maintenance and reduced the complexity of the server scripts.

(iii) It created no security disadvantage: even though the script is downloaded and the names of pages to which the user does not have access rights can be determined, a constructed request for such pages would still be turned away by the server through the cookie-based authentication/authorization system.

These advantages were noted. When it came time to create the account administration system, it was decided that its user interface would be implemented in Javascript on the client. A request for an administration page from the server returns a Web page containing a series of script include statements, a Javascript data structure containing a verbatim copy of the relevant information from the database, and finally a call to a function in one of the included scripts to render the page. No HTML beyond the standard header and footer bars is created by the server.

The user interface to the administration tools is a complex application with buttons for creating, modifying, and deleting accounts; checking username and password fields; displaying the changes to be made in an intuitive fashion; and relaying the necessary instructions to the server once the administrator enacts the changes. Multiple changes can be made to tables of users or groups in the system; the totality of these changes can be seen by the administrator before they submit them in a single transaction.

It would have been a difficult programming task to construct an administration system were server contact to be required after every user action, and the maintenance of state at each stage would make it very complicated. Also, the need for server contact would make the system much slower and unwieldy to work with from the user's point of view.

Detailed testing and evaluation of the site was done some time after the account administration was completed. Many recommendations were made from this, with much of the criticism being directed at the "molecule table" page (from which users control the drug design workflow). This was, at the time, a table containing text, graphics, hyperlinks, and tool tips created entirely on the server. The improvements requested included, among other things, the following:

(i) Help texts, beyond the tool tips, should be available on the same page, removing the need for students to flip between pages. Ideally, these should be context-sensitive.

(ii) Help texts should be illustrated where necessary.

(iii) It should be possible to control how many elements are displayed in the table to make the page quicker to navigate around.

(iv) The tool tips did not work consistently across all target browsers.

(v) The tool-tip display within the browser was outside the designer's control, and the tiny fonts used by default would cause problems for users with visual problems. New tool tips would have to be implemented independently of the browser's built-in mechanism.

(vi) Confirmation should be sought before deleting molecule entries.

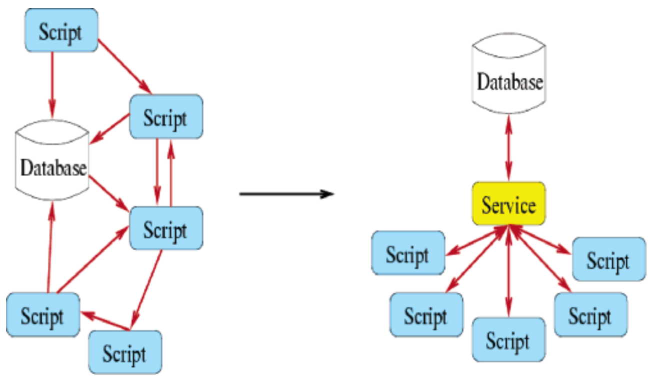


Figure 3. Schematic of system changes. The system has changed from having many scripts (blue squares) interacting with the database (white cylinder) to having them speak to a master service (yellow square), which speaks to the database alone.

(vii) On copying a molecule, a new name should be requested for the copy.

(viii) The page should automatically update itself every few seconds when a job was running, but not otherwise, to limit load on the server.

(ix) A workaround was required for the handling of PNG images in Internet Explorer (IE).

(x) A workaround was required for the handling of CSSs in the same browser.

It, thus, became clear that a major overhaul of the molecule table rendering was required, and that the new engine would need to be much more complicated than what was then in place. The success we had building the complex administration tools with client-side scripting led us to attempt the molecule table rebuild using the same methods.

The script had to build the interactive molecule table screen and implement context-sensitive help, CSS-defined tool tips, popup alerts and queries, automatic intelligent updating, and workarounds for browser foibles. From the outset, this was a much more ambitious project than the administration screens had been.

In any event, with the experience gained from the administration scripts, the implementation went quickly and smoothly, apart from the fixing of the mentioned browser problems, which took considerable time. The results exceeded the expectations of the designers.

Tackling Information Loss. Over the course of development, e-Malaria has had many different formats for storing and transmitting molecule information. As the system developed at the start, file conversions and storage into/retrieval from the database were performed in many different places, and it started to become difficult to keep track of. In particular, it started to become difficult to track down errors caused by information loss, as illustrated in Figure 3.

Eventually, the decision was taken to move most of these disparate conversions and database interactions to a single master service. This would provide one point of contact for information retrieval and submission and would handle all necessary file conversions itself. All interaction with this program would be through standard HTTP methods (GET/POST requests). By doing it this way, we reduced the complexity of the system and broadened its scope; the potential for extending and reusing this element of the code is fairly high.

The internal file format used was developed on the fly and took the form of a simple space-separated column file containing all the information we needed to store, without any self-describing properties. While developing the molecule viewing application, it became clear that Chemical Markup Language (CML)[14] was going to be necessary to communicate partial charges to the Jmol applet. Our initial expectations were that the use of this XML-based file format would take a very large amount of complex programming work. As it turned out, however, the conversion from the internal molecule format to CML could be accomplished using just four (admittedly complex) Perl regular expressions, operating on the file as a whole. This was far less painful than had been expected. Although reading and interpreting a completely general CML format file would be an involved job, reading and writing files which used just a narrowly defined subset of the format (used consistently within the scope of an application suite) would require very little effort. While writing less than general code for CML would lose some compatibility, most of the benefits of the XML/CML approach would be retained, such as self-description and readability by truly standards-compliant applications and also extensibility and future proofing. Most importantly, applications designed to use CML from the ground up could be designed to preserve all information passing through them and generated within them, thus, removing the constraints on the architecture imposed by information loss considerations: either of the schemes shown in Figure 3 could be used without the problems described above. In practice, the left-hand scheme in Figure 3 represents the standard grid computing model, while the right-hand side describes a service implemented through a middleware layer. Our lack of a data standard constrained us to using the latter; for grid computing to work, reliable and extensible data formats such as XML/CML are a necessity.

The realities of the XML/CML approach to information storage are far less problematic than the technical papers describing it seem to imply. While large, complex, and resource-intensive libraries and extensive use of object-oriented code would probably be necessary for writing programs capable of reading *general* CML files, working with a tightly constrained subset of the format requires far less work, and code to do this can be constructed from simple state machines. Such a file format would be readable by general purpose applications because it would be a strict *subset* of CML; local extensions to the internal format could be added transparently, and as long as they adhered strictly to the XML specification, processability by external applications would not be compromised.

If the project were started again, just such a narrowly defined subset of XML/CML would be used internally for data storage. The space efficiency of the format can be raised to quite a high value using appropriate types of tags, and if space were still a problem, compression is a simple and effective solution.

DESCRIPTION OF THE COMPLETED SYSTEM

On first entering the site, the user sees an introduction page containing a picture of a mosquito, a few paragraphs about the disease and this project, acknowledgments, some links, and a list of system requirements for taking part in

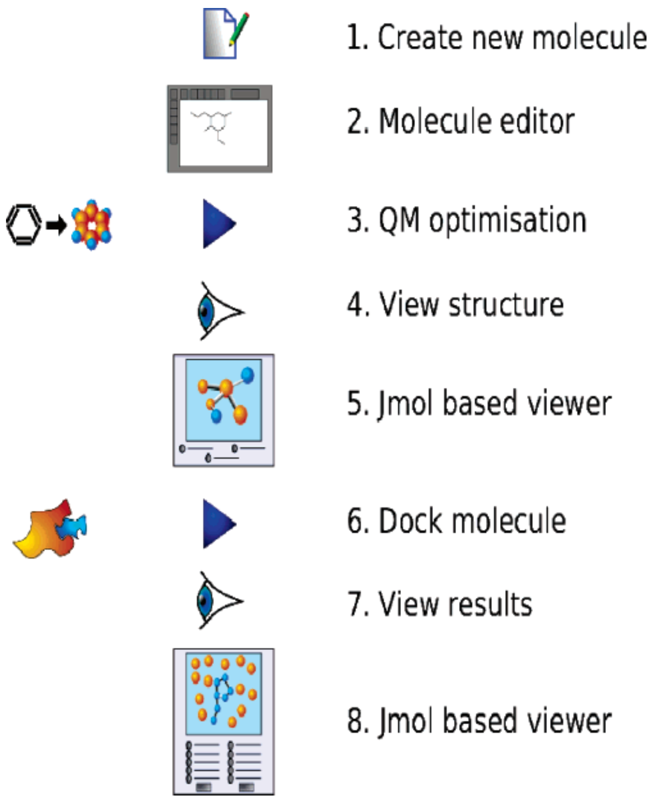


Figure 4. Schematic of e-Malaria user workflow. The molecule editor and the two Jmol-based viewer stages are separate Web-based applications connected together by a central Web-based control panel. In the first design stage, the user clicks "create molecule" and draws their design before returning to the control panel. In the second stage, the user activates the optimization software from the panel then clicks "view structure" to see their molecule in 3D. In the third stage, the user activates the docking software from the panel and then clicks "view results" to see how the molecule fared.

the project [currently tested on Windows (IE and Firefox), Linux (Firefox and Konqueror), and Apple (Safari)]. The IE user interface is slightly different from the others as it turned out that certain browser issues would have taken an unjustifiable effort to work around.

The site has been designed so that anyone with a Web connection can access the teaching material we have prepared without needing an account. A series of links to the teaching material are shown in the navigation bar, sorted under three headings: the project, malaria, and chemistry.

From the point of view of the pupil, the drug design workflow can be thought of as having eight stages, as shown in Figure 4.

When a student starts, the molecule table screen will show the "create new molecule" icon (1), along with a dialogue explaining what is going on and prompting them to click on the icon. Doing this will take them to the molecule editing page (2), where they can construct a molecule through freehand drawing, or by using a library of premade molecule fragments. On submitting their molecule from this page, the molecule is entered into the system along with a geometry optimization/energy minimization job. Clicking the activate button (3) puts the job into a queue for execution. When compute resource becomes available, a two-stage minimization process is run to obtain a reasonable three-dimensional structure for the molecule along with atomic point charges. Once finished, the results are entered into the molecule database alongside the initial 2D structure entered by the student. The user is then prompted to examine the calculated 3D structure and charge distribution (4 and 5) and then activate a docking job (6). Docking in this context means finding an energetically good geometric fit for the molecule in the active site of the target enzyme. After docking has finished, the best structure obtained is inserted into a results table on the same page as the molecule table. The user is encouraged to look at the energy numbers obtained from the docking calculation and then click on the view icon (7) to examine the results in a 3D display application (8).

HARDWARE SETUP AND FIREWALL

e-Malaria depends on a number of components sited on different machines working together in a seamless manner. The Web server and the primary database are sited on a dual Xeon machine, "Green", with approximately 1 terabyte of RAID disk space; a second similar machine, "Purple", has been set aside for the United Devices subsystem which manages the multiple remote machines that perform the docking jobs. A further pair of machines have also been set aside for local execution of docking and quantum mechanics jobs. Finally (at the time of writing), a low-powered workstation PC has been put in place to act as a firewall. Further computing power for running docking jobs will be made available from the University's Science Learning Centre.[15] This system is shown in Figure 5.

There are two main problems with the above architecture:

(i) Input/output bandwidth is limited to 100 Mbit/s by the network cards in the firewall.

(ii) Only one machine (the Web server) is made externally visible; the UD server cannot be accessed by external machines, limiting the UD system extensibility to entities behind the university's central firewall.

There are plans to upgrade the firewall and make the UD server visible once security concerns have been satisfied.

Firewall. The e-Malaria project requires a server that is visible outside Southampton University's firewall. This obviously creates a potential security hazard. To make any machine available to the outside, university regulations require its managers to take responsibility for any security issues. To protect against certain types of breaches, it was decided to implement an internal firewall solely for the projects' server.

While the university's firewall will protect us from the bulk of the service scanning attacks taking place over the Internet, what it cannot protect us from is errors or omissions in the security of the publicly exposed software running on the server machine. Of particular worry are the Apache Web server, the database, and, critically, the CGI programs operating the various public services provided on the server. Even with the most dedicated patching regime, the possibility of the machine being compromised remotely cannot be ignored. If this were to happen, not only would the data and programs on the machine be at risk (hazardous to the projects) but, much more importantly, the machine could be used as a base for further attacks on other machines within the university firewall. To mitigate the risk of a compromised machine being used as a springboard for launching attacks

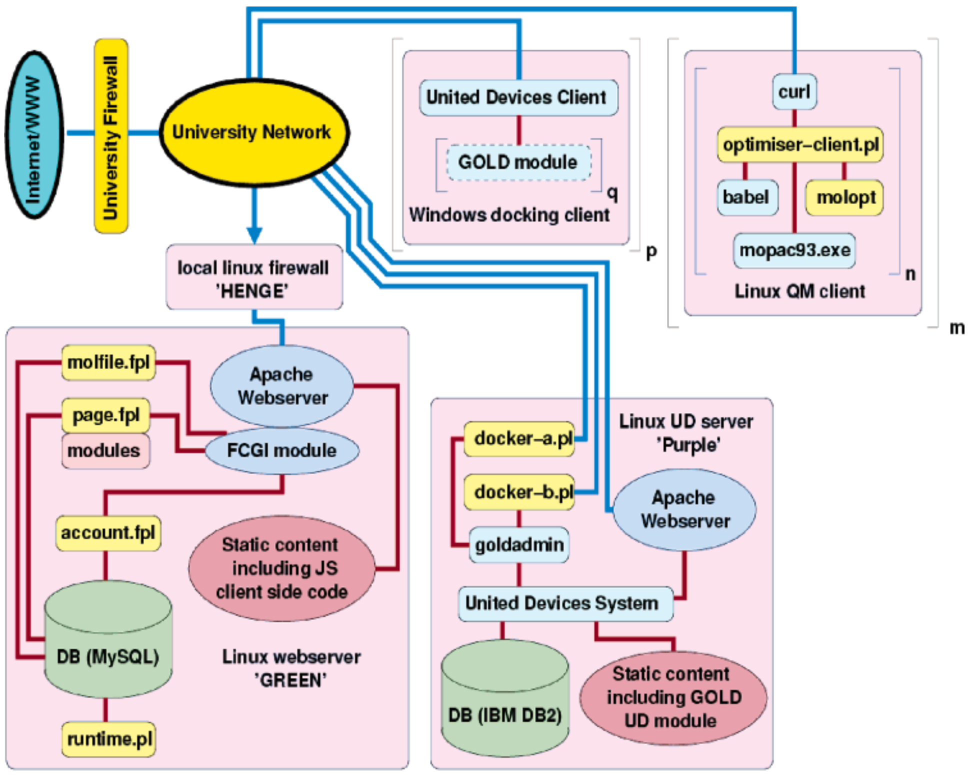


Figure 5. Schematic block diagram of the e-Malaria system. Curved pink boxes show machines; curved yellow boxes are software components written for the project; curved pale blue boxes are externally written software components; green cylinders are databases; the Apache Web server and its FCGI module are shown as pale blue ovals. Labeled square brackets indicate opportunities for application- and machine-level parallelization. Red lines indicate connections between software elements running on the same machine, while blue lines are connections between software elements running on different machines.

on other university machines, it was decided to put a firewall into place to block connections *outbound* from the server.

MOLECULAR DYNAMICS/QUANTUM MECHANICS (MD/QM) OPTIMIZER

Users of e-Malaria construct their molecules in a 2D graphical editor. Unfortunately, the editor does not constrain the bond lengths or angles or the element types or bond orders of the molecules entered or derive realistic two- or three-dimensional structures from it. Molecules created thus need considerable checking and modification before they can be used for docking purposes. This preparatory phase, though in reality consisting of many separate stages, is referred to as the “MD/QM optimizer” throughout this paper.

The QM optimizer is implemented in a client−server fashion; one or more instances of the client script running on a remote Linux machine poll the Web server machine with requests for jobs. All network traffic is performed using the HTTP protocol as implemented in the “curl” application. If no job is received from the server, the client will wait for a set period before polling again. If the server does not respond to the request, the client will retry a set number of times before pausing in “standby” mode for an hour. The client code has been written in such a way that multiple servers may be polled by the same script; this is to facilitate reusability, it being a straightforward task to add other servers to the system and so providing a valuable service to other projects. Any errors which occur during the optimization process are communicated back to the server through the same type of Web request transfer as that of a normal result. Most commonly occurring error conditions are simple user

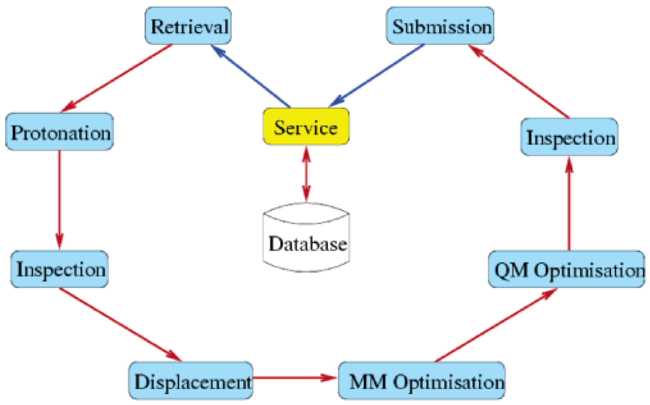


Figure 6. MD/QM optimization process. Molecule data is retrieved from the database and sent out to an optimization service. From here, it is protonated, inspected for simple errors such as valence problems, subjected to a small displacement to avoid nullity problems, and its bond lengths corrected with a molecular mechanics force field. Then, the geometry of the molecule is optimized using a semiempirical quantum mechanics method and the results inspected for bonding rearrangement, and then the finished molecule is returned to the database.

mistakes such as incorrect valencies and inorganic element types. More complicated problems are also checked for, though, particularly rearrangement of the bonding pattern of the molecule during the quantum mechanical procedure. Molecular topology is calculated both before and after this calculation and the results compared. The MD/QM process is illustrated in Figure 6.

On completion of a successful download of a molecule structure file, the first processing step is protonation; an open source program called “OpenBabel” is used to do this. After the molecule has been protonated, we would ideally like to move straight to the QM calculation to determine the accurate geometry. Unfortunately, we cannot do this because the QM software packages available to us cannot optimize a structure from the simple topology data created by our editor application. As a rule, QM packages can interpret either *Z*-matrix or Cartesian data, and writing software to automatically and reliably generate a good *Z* matrix for arbitrary molecules is difficult, which leaves us with the Cartesian coordinates of a molecule. Where these Cartesian coordinates have major errors in bond lengths, the QM software will simply not consider them to be bonded. A covalent bond is not a quantum mechanical observable and any bonding effects must arise directly from the electron distributions calculated. A program was therefore written to do a rough geometric optimization using a simple molecular mechanics force field. This program is called “molopt”. The objective of molopt was not to determine the correct structure but to move the atoms of the molecule such that the interatomic distances of bonded atoms are sufficiently close for a bond to arise and nonbonded atoms are sufficiently separated so that no bond can arise. Molopt is a C program written for the e-Malaria project to solve the optimization problem described above. It also performs a variety of error checks and calculations on the initial molecule configuration—verifying the chemical “sanity” of the user’s molecule:

(i) The number of atoms is counted; molecules with more than 100 atoms are rejected.

(ii) Inorganic elements are excluded, to avoid problems with the limited parametrization of the semiempirical model.

(iii) The formal charge on the molecule is calculated from the element valences; this is needed by the QM software.

(iv) The number of distinct molecules in the file is calculated through a mark and sweep algorithm designed for arbitrary molecule topologies. If more than one molecule is present in the file, the file is rejected.

(v) Bond counts and orders of atoms are checked against valencies. In the case of aromatics, carbon is allowed to bond to exactly three other atoms, while other elements can bond up to their normal full valence number to cover borderline cases such as imidazole and furan. It must be emphasized that bonding and bond order are not quantum mechanical observables; this makes the situation complicated, and some sort of compromise had to be reached.

Before the optimization is performed, the coordinates of the molecule are displaced by a small random distance to remove degeneracy problems for molecules whose atoms are all at the origin (null coordinates). This allows the optimizer to generate structures for molecules for which the user only has topology information, such as an InChI code[16] or SMILES string. Many databases only contain such limited data, and a service for obtaining meaningful structures for these on request would have many applications. Although not needed for the purposes of e-Malaria, this ability may make the code useful for other projects within other programs. Also, before optimization begins, stereochemical data from the bonding section of the input file is incorporated at a basic level. Molecules created by the molecular editor are bound to the plane $Z = 0$. Bonds described as “upward wedges” in the file have a constant subtracted from their *Z* value, while those described as “downward wedges” have the same number added to their *Z* value. This approach will probably fail for some molecules with unusual topology and certainly fail in the cases described above where no initial coordinates at all have been specified. For simple cases where a more-or-less planar molecule has been sketched containing unstrained stereocenters, this approach works quite well, though. Correct generation of molecules with no coordinates and multiple stereocenters is a complex problem and beyond the scope of the current project.

Once these checks and calculations have been made, a steepest-descent optimization is performed on the molecule’s atomic coordinates using a very simple force field. The parameters for the field are specified in a file, and where no parameters are available, sensible defaults are used—we are not trying to determine an exact structure with this force field. The parameters for this force field are listed in the Supporting Information for this paper, based on information published elsewhere.[17]

A more accurate geometry opimization using a QM package is then performed. During the early development of the software, the program Gaussian 98 was used, for which the university has a site license. This license explicitly forbids providing any sort of service that makes use of their software, though, and so another program had to be found before rollout. After much searching and trial and error, a version of MOPAC[18] was obtained which compiles correctly under Linux. Several other MOPAC versions had been tried, but these met problems with modern FORTRAN compilers. Early versions of MOPAC were in the public domain, but the software was eventually acquired by Fujitsu corporation. In light of this, we were obliged to obtain an updated but working copy of the last public domain version of the software. Our modular approach to design means that, if we were to create further systems based on e-Malaria for internal use, we would, for example, be able to substitute Gaussian seamlessly where licensing permits.

After MOPAC has finished, the topology of the structure it has produced is compared against that of the original structure, to check for the molecule breaking apart.

While we have used the standard HTTP protocol for internal communications between the MD/QM clients and server, it would be a simple matter to encrypt the traffic using HTTPS. Secure transfers between machines on the local network were considered unnecessary for the project. They might become necessary should a system based on e-Malaria be rolled out to other sites, or if an internal system were to be used in the context of an industrial contract with tighter information security requirements.

DOCKING CALCULATIONS

Docking calculations have been implemented using the genetic-algorithm-based GOLD software package[19,20] generously provided for the project by the Cambridge Crystallographic Data Center. Gold uses a simple force field to calculate docking affinity as a function of inter- and intramolecular (van der Waals) forces and hydrogen bonding. A pool of molecular conformations is maintained throughout the calculation, and the docking “fitness” of each is evaluated at every step. The conformations are “bred” with each other in a genetic algorithm to maximize docking fitness; small changes are made to the conformations of molecule confor-

Table 1. Statistics on the State of the e-Malaria System on August 18, 2005

groups	16
sites	5
users	85
molecules	692
average minimization time	9 s
average docking time	5 min 20 s

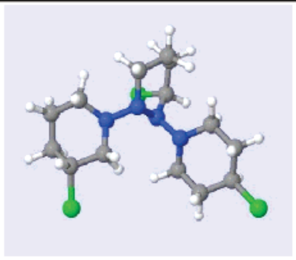
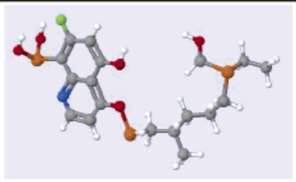
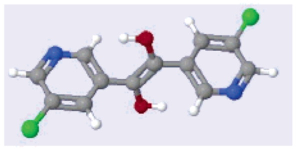
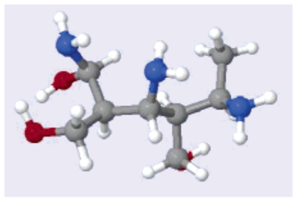
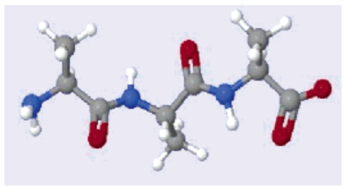
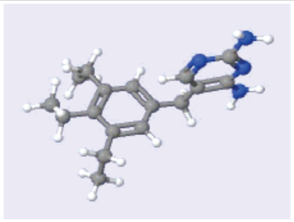
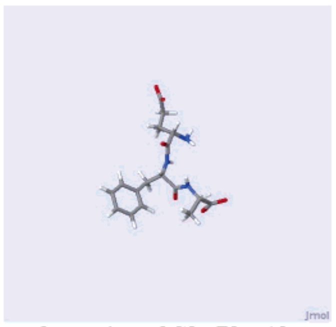
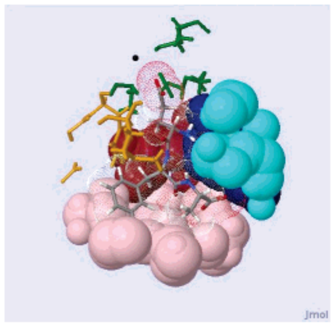
mation at each stage. Ligand bond lengths are fixed, but torsion angles are movable; the protein is mostly fixed, although hydrogens bonded to nitrogen and oxygen are allowed to move in a limited way. To obtain a license for this software, we needed to keep it secure from theft. To do this we used the UD platform.

Our target system was obtained by taking a 16 Å spherical scoop from the active site of the DHFR part of the bifunctional malarial enzyme. A fragment of the NADPH cofactor is present, along with a water molecule at the top of the site found present in the crystal structure. The protein structure we have used is the 1J3I structure[8] from the Brookhaven Protein Databank—wild-type *Plasmodium falciparum* dihydrofolate reductase-thymidylate synthase (PFDHFR-TS) complexed with WR99210, 3NADPH, and DUMP.

UNITED DEVICES

United Devices produces software for managing large-scale distributed computation systems.[21] Their platform follows a client—server model. Although it is possible to

Table 2. Some Images of Molecules Taken from the e-Malaria System on August 18, 2005

 **Many linked nitrogens, score 33.25**	 **Organo-phosphorous molecule, score 68.71**
 **Symmetrical double ring system, score 41.27**	 **Substituted, branched alkane, score 30.02**
 **Ala-ala-ala tripeptide, score 50.15**	 **Trimethoprim, score 57.49**
 **Docked conformation of Glu-Phe-Ala, score 68.88**	 **Image of Glu-Phe-Ala docked into active site taken from web visualiser. The tripeptide is shown in wireframe with a dotted surface and CPK colourscheme; the active site is shown partly as solid van der Waals spheres and partly as wireframe and highlighted in bright colours to ease perception. Colour, representation and orientation can all be controlled from the web application.**

build a server system comprised of multiple machines, communicating with each other through Web services, this was deemed unnecessary for our relatively small-scale operation. Instead, the UD software, along with the IBM DB2 database and Apache server that come with it, were installed on the dual Xeon server; the system is fairly resource hungry, and little else could profitably be run on this system.

The client part of UD can be obtained for both Linux and Windows systems. While most of the machines over which we had direct control run the Linux operating system, most of the machines available for remote calculation within our organization ran Windows; for the purposes of obtaining the most processing capability for the least system management, a standard Windows installer executable with the IP address of the server built into it was produced; after installation, this client repeatedly polls the server requesting jobs. All transactions are done using standard HTTPS. The first time a docking job is sent to the client, a copy of the GOLD executable tailored for execution in the UD environment is sent along with it. Subsequent jobs use the cached copy. To maximize the security of the system, the executables we are using are cryptographically signed by the UD system to verify their authenticity. This is important in maintaining the security of the underlying docking application code

PERFORMANCE AND STATISTICS

Tables 1 and 2 describe the state of the e-Malaria system on August 18, 2005 and show a selection of molecules created by the system's users.

CONCLUSIONS AND FUTURE DIRECTIONS

This project has created a novel Web-based chemistry teaching tool for schoolteachers to experiment with. So far, the feedback obtained has been extremely positive and indicates that the participating teachers understand the potential of this approach.

All the objectives of our original project specification have been met, and some have been exceeded; the extension to testing arbitrary molecules is a particularly important achievement.

A number of enhancements to the site and the machinery underpinning it have been proposed, including "unwrapped" schematic diagrams illustrating the hydrogen bonds present between the molecule and active site, visualization of the molecular orbitals obtained from the QM calculations (using a development version of the Jmol applet), and also a reimplementation of the simplified peptide-based molecular design application for younger students to use. A further-reaching modification would be to provide a more sophisticated degree of control over the apparatus, enabling research workers to use the system, either at a component level or in an integrated fashion, for simplifying and automating various jobs such as obtaining force field parameters for arbitrary small molecules. When automated systems are designed in a consistent way, it becomes easy to extend them by adding further tasks and interfaces. The network-service-oriented design of the e-Malaria component systems enables application of its components to more general problems and easy extension to other projects across institutional barriers.

A further piece of research that would be straightforward to automate using the currently built system is the screening of all tripeptide molecules against the malarial DHFR target, and possibly also the human variant.[22] The high scores for the tripeptides shown in the Performance and Statistics section are interesting and warrant some further investigation. Structures for the tripeptides with correct topology but limited geometric content could be created by a script and submitted into the system for automated geometry correction and docking. Such a script could be executed remotely, with interaction with the system being purely through the existing Web interface.

The authors feel that e-Malaria is one of the first complex systems designed from the ground up to use advanced, research-level chemical modeling software for the teaching of chemistry. By building a system that follows part of the workflow used in real-world drug design and employing the same tools used by real-life scientists in this endeavor, students can gain an insight into how the scientific method is applied in practice.

ACKNOWLEDGMENT

We thank the CCDC for the generous use of their GOLD docking software; JISC and the EPSRC for funding (EPSRC Combechem Grant GR/R67729); John Metcalfe; John Frampton, and his students at Havant College; Lee Banting and Matthew Ellis and their pharmacology students at Portsmouth University. We thank Chris Curtis, of the London School of Hygiene and Tropical Medicine, for providing us with the photograph of a mosquito of the correct type used in Figure 1 and on the front page of the site.

Supporting Information Available: A configuration file describing the docking calculations performed using the GOLD software, details of the MOPAC parameters used in the molecular geometry optimizer, tables of force field parameters, also used in the optimizer, and screenshots of the Web site. This information is available free of charge via the Internet at http://pubs.acs.org

REFERENCES AND NOTES

(1) e-Malaria Project Home Page. http://emalaria.soton.ac.uk (accessed August 30, 2005).

(2) Tai, K.; Murdock, S.; Wu, B.; Ng, M. H.; Johnston, S.; Fangohr, H.; Cox, S. J.; Jeffreys, P.; Essex, J. W.; Sansom, M. S. P. BioSimGrid: towards a worldwide repository for biomolecular simulations. *Org. Biomol. Chem.* **2004**, *2*, 3219−3221.

(3) Brailsford, T.; Burt, C.; Calder, J.; Davies, M.; Edge, C.; Murray-Rust, P.; Overington, J.; Richardson, C. M. Virtual Education for Medicinal Chemistry: Early Experiences in Industrial-Academic Partnership for Continuous Learning. *Internet J. Chem.* **2001**, *4*, 3.

(4) Anderson, D. P.; Cobb, J.; Korpela, E.; Lebofsky, M.; Werthimer, D. SETI@home: An experiment in public resource computing. *Commun. ACM.* **2002**, *45* (11), 56−61.

(5) Anderson, D. P. BOINC: A System for Public-Resource Computing and Storage. 5th IEEE/ACM International Workshop on Grid Computing, November 8, 2004, Pittsburgh, PA, U. S. A.

(6) Prix, R.; Pössel, M.; Machenschalk, B. Gravitationswellen im Heimcomputer: Einstein@home. *Astron. Heute* **2005**, September 14.

(7) Allen, M. Do it yourself climate prediction. *Nature* **1999**, *401*, 642.

(8) Richards, W. G. Virtual screening using grid computing: the screensaver project. *Nat. Rev. Drug Discovery* **2002**, *1*, 551−555.

(9) Yuvaniyama, J.; Chitnumsub, P.; Kamchonwongpaisan, S.; Vanitchtanankul, J.; Sirawaraporn, W.; Taylor, P.; Walkinshawl, M. D.; Yuthavong, Y. Insights into Antifolate Resistance from Malarial DHFR-TS Structures. *Nat. Struct. Biol.* **2003**, *10* (5), 357−365.

(10) Jmol Home Page. http://jmol.sourceforge.net/ (accessed August 30, 2005).

(11) ACD/Structure Drawing Applet. http://www.acdlabs.com/products/java/sda/ (accessed August 30, 2005).

(12) Krause, S.; Willighagen, E.; Steinbeck, C. JChemPaint—Using the Collaborative Forces of the Internet to Develop a Free Editor for 2D Chemical Structures. *Molecules* **2000**, *5*, 93−98.

(13) ChemAxon Home Page. http://www.chemaxon.com (accessed August 30, 2005).

(14) Murray-Rust, P.; Rzepa, H. S. Chemical Markup Language and XML Part I. Basic principles. *J. Chem. Inf. Comput. Sci.* **1999**, *39*, 928−942.

(15) Science Learning Centres Portal Home Page. http://www.science-learningcentres.org.uk (accessed August 30, 2005).

(16) Stein, S. E.; Heller, S. R.; Tchekhovskoi, D. An Open Standard for Chemical Structure Representation: The IUPAC Chemical Identifier. *Proceedings of the 2003 International Chemical Information Conference*, Nimes, France, October 19−22, 2003; Infonortics: Tetbury, U. K.; pp 131−143.

(17) (a) Huheey, pps. A-21−A-34. (b) Cottrell, T. L. *The Strengths of Chemical Bonds*, 2nd ed.; Butterworths: London, 1958. (c) Darwent, B. D. National Standard Reference Data Series, National Bureau of Standards, No. 31, Washington, DC, 1970. (d) Benson, S. W. *J. Chem. Educ*. **1965**, *42*, 502. The above are cited by the Web page http://chemviz.ncsa.uiuc.edu/content/doc-resources-bond.html (accessed August 30, 2005).

(18) Stewart, J. J. P. Optimization of Parameters for Semiempirical Methods III—Extension of PM3 to Be, Mg, Zn, Ga, Ge, As, Se, Cd, In, Sn, Sb Te, Hg, Tl, Pb, and Bi. *J. Comput. Chem.* **1991**, *12*, 320−341.

(19) Jones, G.; Willett, P.; Glen, R. C.; Leach, A. R.; Taylor, R. Development and Validation of a Genetic Algorithm for Flexible Docking. *J. Mol. Biol.* **1997**, *267*, 727−748.

(20) Verdonk, M. L.; Cole, J. C.; Hartshorn, M. J.; Murray, C. W.; Taylor, R. D. Improved Protein−Ligand Docking Using GOLD. *Proteins* **2003**, *52*, 609−623.

(21) Uk, B.; Taufer, M.; Stricker, T.; Settanni, G.; Cavalli, A.; Caflisch, A. Combining Task- and Data Parallelism to Speed up Protein Folding on a Desktop Grid Platform—Is efficient protein folding possible with CHARMM on the United Devices MetaProcessor? *Proceedings of the CCGRID 2003, IEEE International Symposium on Cluster Computing and the Grid*, Tokyo, Japan, May 2003.

(22) Klon, A. E.; Heroux, A.; Ross, L. J.; Pathak, V.; Johnson, C. A.; Piper, J. R.; Borhani, D. W. Atomic Resolution Structures of Human Dihydrofolate Reductase Complexed with Nadph and Two Lipophilic Antifolates. *J. Mol. Biol.* **2002**, *320* (3), 677−693.

CI050383Q