Web Service Infrastructure for Chemoinformatics

Xiao Dong, Kevin E. Gilbert, Rajarshi Guha, Randy Heiland, Jungkee Kim, Marlon E. Pierce, Geoffrey C. Fox, and David J. Wild*

Indiana University School of Informatics, Community Grids Laboratory, and Chemical Informatics and Cyberinfrastructure Collaboratory, Indiana University, 901 East Tenth Street, Bloomington, Indiana 47408

Received October 10, 2006

The vast increase of pertinent information available to drug discovery scientists means that there is a strong demand for tools and techniques for organizing and intelligently mining this information for manageable human consumption. At Indiana University, we have developed an infrastructure of chemoinformatics Web services that simplifies the access to this information and the computational techniques that can be applied to it. In this paper, we describe this infrastructure, give some examples of its use, and then discuss our plans to use it as a platform for chemoinformatics application development in the future.

INTRODUCTION TO WEB SERVICE TECHNOLOGIES

Recent advances in Web technology enable computational tasks to be carried out remotely in a straightforward manner through standard Web protocols. By making simple Web services and Web application programming interfaces (APIs) available, sites such as Google Maps[1] have fueled a wave of innovation in application development, particularly in "mashups"—quickly developed desktop or Web applications which use multiple services or data sources. Also, standards have been developed for the communication of data, metadata, and meaning (including XML[2] and derivative markup languages and languages that describe ontologies such as OWL[3]) and the access of applications remotely (Web Services, including Simple Object Access Protocol - SOAP,[4] Web Service Description Language - WSDL,[5] and Uniform Discovery, Description and Integration - UDDI[6]). These technologies are considered part of the next wave of Internet usage known as the *Semantic Web*.[7,8] Additionally, the term "Web 2.0" is being used to describe the use of the Web in a social, collaborative sense.

The Murray-Rust group at Cambridge[9] has carried out some interesting research in the use of these techniques in chemoinformatics. The use of XML-derived standards for chemistry and chemical informatics applications including the development of Chemical Markup Language (CML), an XML schema for chemistry, and applications of RSS, has been documented in a series of papers by Peter Murray-Rust and Henry Rzepa et al.[10−15]

Web services are an emerging way of aggregating and integrating data sources and software. They allow software applications and data sources to be published on the Internet (or on intranets), thus making tools and data widely available with a standardized interface and facilitating the construction of applications that employ distributed resources and data to solve complex tasks. Three standards have emerged for creating Web services: WSDL is an XML-based standard for describing Web services and their parameters; SOAP "wraps around" existing applications to map abstract interfaces in WSDL to their actual implementations; and UDDI effects the publishing and browsing of Web services by user communities. The idea of using Web services is just beginning to take hold in the life sciences.[16] Much of the initial groundwork has been in bioinformatics (to the extent that several service providers such as the EBI[17] now offer their tools as Web services). There has been less adoption in chemoinformatics.

Outside of the life sciences, some vendors have gone further and created Javascript wrappers which enable their services to be called extremely easily from a Web page. The Google Maps API, for instance, can be imported into a Javascript section of a Web page and then calls can be made to functions which access the map services such as map drawing, determination of address longitude and latitude, and placing markers on maps. A typical Google Maps mashup will read information including longitude and latitude coordinates or addresses from a server database (often MySQL through PHP) and will display this information on a map using the maps Javascript API. Numerous such mashups are available on the Web.[18]

WEB SERVICE INFRASTRUCTURE

At Indiana University, we have created (and are continually expanding) an infrastructure of Web services for chemoinformatics. The Web services currently available are mostly based on Java. Though there is no reason why the services could not use other languages, many of the underlying tools (such as the CDK for chemoinformatics functionality) are written in this language. Another reason for the use of Java is that it allows us to deploy our Web services in a Tomcat application container, which allows us to easily maintain a variety of services and provides a high level of integration with our development environments. The services themselves are hosted on a variety of servers and in general are separate from database and computational functionality. Thus Web services that provide database functionality will contact a remote database server to retrieve results. Our current setup is based on Tomcat 5.5 with the services themselves being written using Java 1.5.0.

* Corresponding author e-mail: djwild@indiana.edu.

10.1021/ci6004349 CCC: $37.00
Published on Web 06/29/2007

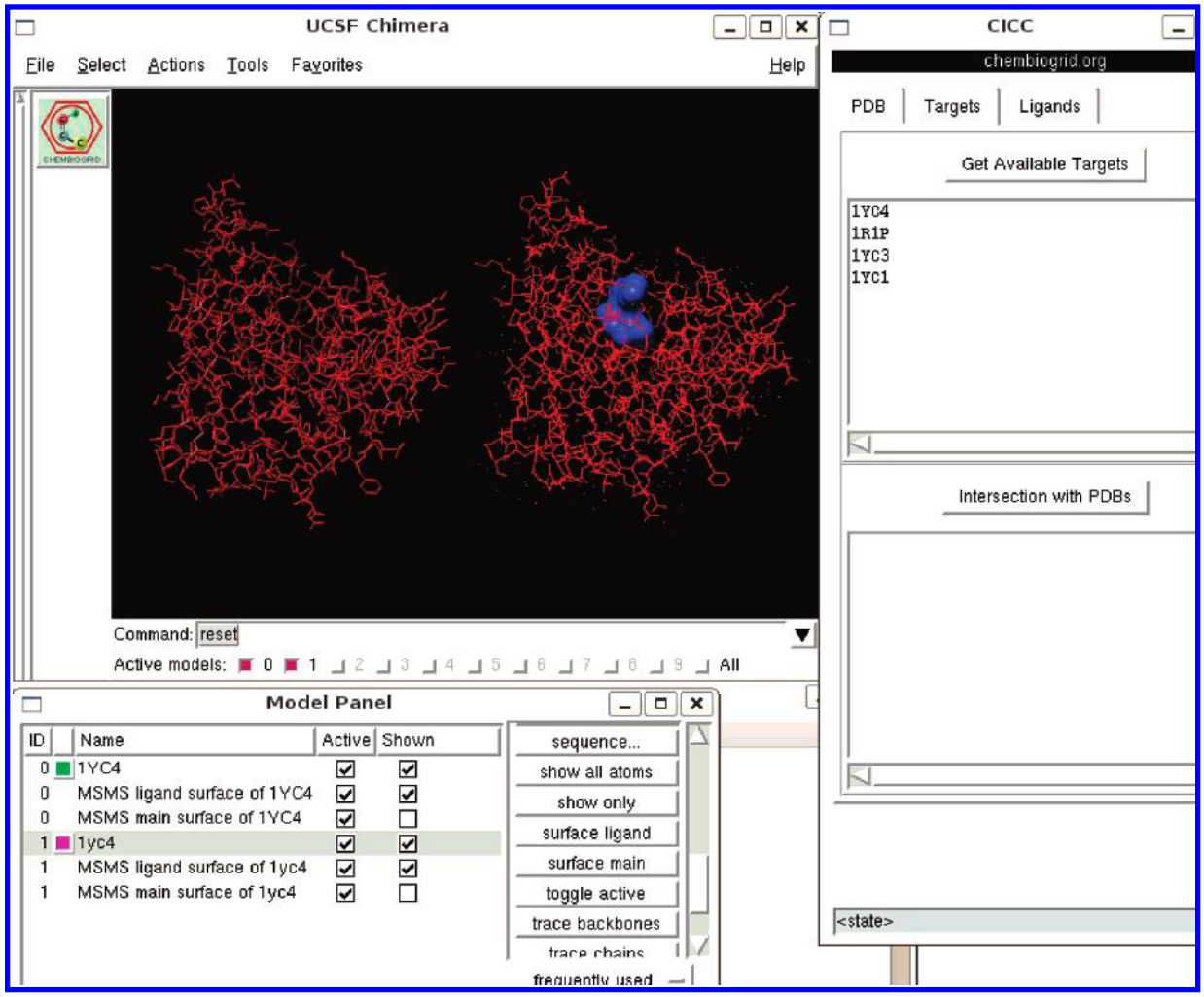

Figure 1. A screenshot of the PubDock Chimera interface. Through scripting, the Chimera program makes calls to our services to retrieve docked structures and scores.

Given a Web service implementation, one must be able to communicate with it to be able to do anything useful. The distributed computing community has described a number of approaches to the problem of serialization of data and messages for the purposes of communication in distributed environments. For our services, we have focused on the use of the SOAP standard for Web service communications. This protocol is an XML formatted document that is passed over HTTP from the Web service client to the Web service and vice versa. It should be noted that the use of SOAP requires that messages and data must be encoded and decoded in a specific format. This functionality is provided by the AXIS libraries,[19] and our system uses version 1.6. Using AXIS means that the Web services themselves do not need to be specially designed to be services: they are written in the form of ordinary Java classes and methods. The Web service layer is handled by the AXIS libraries, which accept a SOAP message, decode it to extract the relevant function arguments, call the appropriate Web service classes, and finally encode the return value into a SOAP document for return to the caller (it should be noted that there is an alternative framework, Xfire,[20] available for handling SOAP processing, but this is incompatible with AXIS). As a result construction of the Web services is made very easy. When large amounts of data must be passed to and from a service, it can be quite inefficient if a SOAP encoding must be generated: not only must the large object be converted to a textual representation but also the actual process of encoding and decoding can become time-consuming. In these cases, it is beneficial to pass *links* to the actual data that are sent to a service or returned from a service. More formally, services can be written such that they accept a *universal resource indicator* (URI), which might resolve to a file on the local disk, a remote HTTP URL, or even an FTP site. Thus rather than sending SOAP encoded data to the service, we simply send the URI of the data to the service. The service will then access the data from the specified URI. Similarly, when the (possibly) large result is available, the service can return a URI to the result data, which the client can then download. In both cases, the memory and speed requirements for the communication are significantly lower.

Given that we provide a number of different Web services, how does one identify what is available? This is an open problem in the Web service development community in general, and a number of approaches have been proposed and implemented. One of the more well-known approaches is to create a UDDI registry. As a result, rather than keep track of multiple Web services in different locations, one can now keep track of the location of a single registry which can be queried to identify relevant Web services, their capabilities and locations. Our initial approach has been to collect the links to the WSDL for each Web service on a

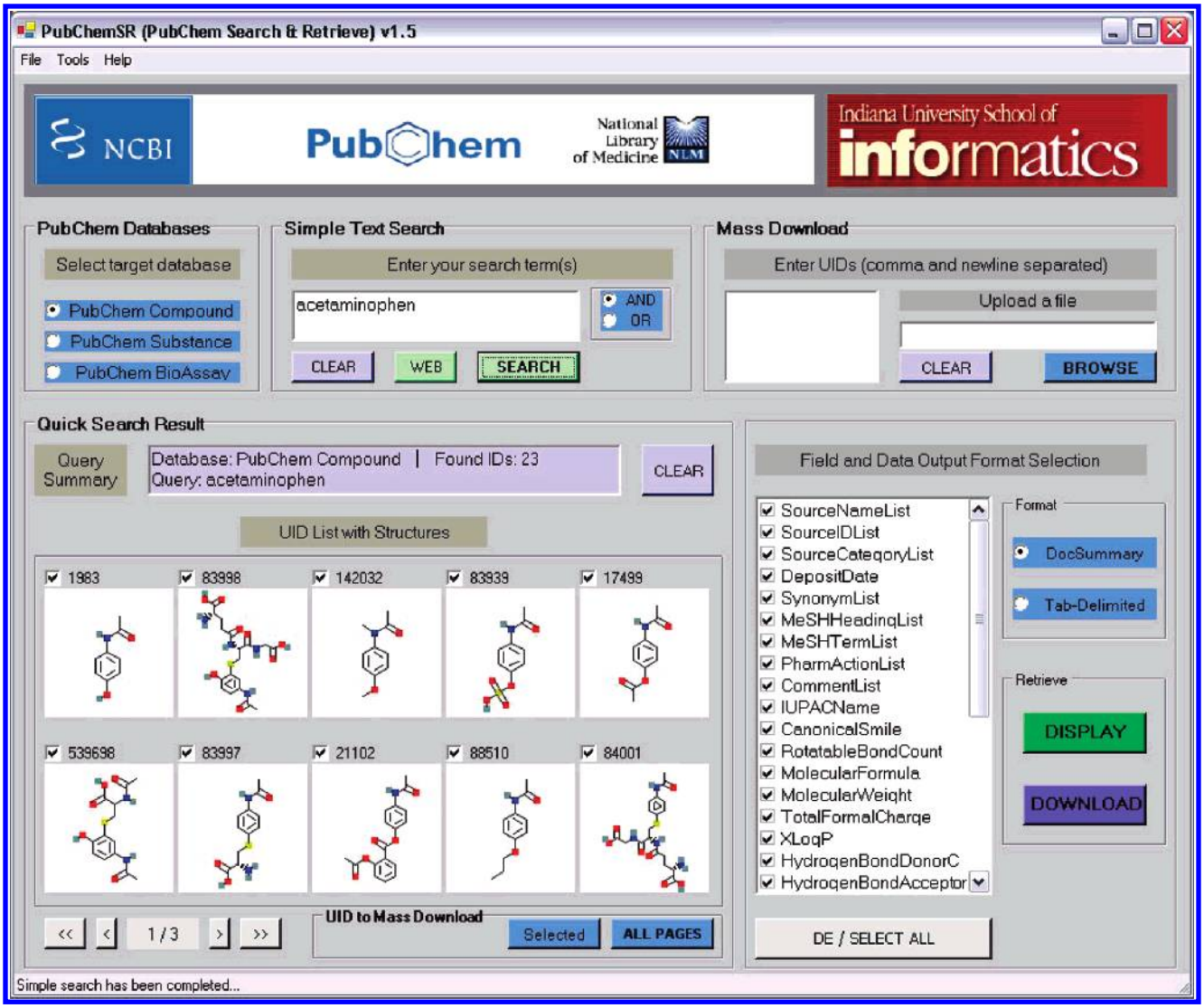

Figure 2. The PubChemSR desktop program for accessing PubChem and integrating with Excel.

simple HTML page. This is certainly easy to use but is not necessarily amenable to machine processing. Thus we have also created a UDDI registry within which our Web services are being aggregated. Both these approaches require us to keep track of the status of either resource—a Web page in one case or a registry in the other. An alternative approach is to have Web service related information *pushed* to users and/or automated systems. To achieve this, we additionally provide an RSS feed which lists all available Web services along with their WSDLs. Thus, one can keep track of the RSS feed, either in a program or with the use of ones favorite RSS reader, and any time a new service is available, it will be marked in the RSS feed. The RSS feed of Web services also leads to other interesting possibilities related to mashups, where the information in the RSS feed is processed and possibly combined with information from other sources to generate a new RSS feed.

Though our approach is quite flexible and provides a significant amount of functionality, there are a number of problems. Because of our focus on the use of Tomcat as a hosting environment, our Web services must currently be written in Java. As a result, though a certain program may not require Java at all, a Java wrapper must be written for it to be incorporated into our system. Another problem is the use of SOAP for communication with services. Though the protocol is relatively well defined, a number of aspects remain ambiguous, and, as a result, different libraries may not always be able to process a SOAP message in the same way. An example of this is for a Web service that accepts or returns a 2D array of integers. On the service side, the AXIS libraries will handle this situation. However on the client side, current SOAP libraries for PHP and Python cannot process 2D arrays, and thus clients cannot be written in these languages to use the Web service. As a result of this problem, we are in the process of investigating other formats, which avoid the ambiguities and incompatibilites in SOAP implementations.

The Web services we have created so far are categorized below. Except where noted, all our Web services are available to be used by other members of the academic community. A full, regularly updated list of services is available at http://www.chembiogrid.org/projects/proj_ws_all.html, including hyperlinks that allow the services to be accessed.

Database Services. We maintain a Linux server running the PostgreSQL database system, with gNova CHORD[21] installed to allow chemical structure searching. Several databases are exposed through Web service wrappers, in

particular we maintain a copy of the NCI Developmental Therapeutics Program Human Tumor Cell Line data set which contains approximately 200 000 compounds, around 40 000 of which have associated screening results for 60 tumor cell line assays. We are currently using this database as a surrogate for high-quality high throughput screening data. We also keep a local "sandbox" copy of the PubChem database which is chemical structure searchable through the Web service interface and which is regularly updated. In addition we have a database of 3D structures of PubChem compounds, a database of these 3D structures docked into a variety of proteins, a database of chemical structures extracted from PubMed abstracts, a database of derived properties for PubChem compounds, and a quantum chemistry database.

Chemoinformatics Computation Services. We have created Web service wrappers around several free and commercial chemoinformatics tools. The commercial tools that we have generously been permitted to use by OpenEye Inc.[22] and Digital Chemistry Ltd.[23] are currently only available within the Indiana University environment and include OpenEye FRED (for docking), OMEGA (for 2D to 3D conversion), FILTER (for property calculation and filtering), and Digital Chemistry Divisive K-Means (for clustering). We have a close working relationship with the Murray-Rust group in the Unilever Center for Molecular Informatics at Cambridge University[24] that is one of a small number of sites that has pioneered semantic Web approaches to chemoinformatics. We have implemented several of their Web services locally, including InChIGoogle, InChIServer, CMLRSSServer, and OSCAR[25] (for automatic mining of chemical structure information from documents). We have also implemented a large amount of the functionality of the Chemistry Development Kit[26] (CDK) as Web services, including descriptor calculation, 2D similarity and fingerprint calculations, and 2D structure depiction. Finally, we have implemented a special modified Web service implementation of ToxTree[27] for toxicity flagging.

Statistical Services Based on R. A variety of services relating to the R statistical package[28] has been implemented. These include linear and nonlinear regression services (such as feed forward neural networks and random forests), classification services (linear discriminant analysis and random forests), and clustering services (k-means) as well as a variety of miscellaneous services such as t-tests, sampling distribution, and 2D scatter and bar plots. In addition to providing model development services, we are able to present arbitrary R code as Web services. This allows us to provide access to prebuilt predictive models as Web services. Examples include predictive models for toxicity, mutagenecity, and cancer activity. Since many of these services are capable of handling large data sets, they have been written so that one can send in the data directly (via a SOAP encoded message) or a link to the data. Currently, if a link is specified it is required to be an HTTP URL. Future upgrades will allow services to utilize arbitrary URIs.

EXAMPLE APPLICATIONS

We have created several applications that use our Web services, to demonstrate the utility of the Web service approach in making it easy to develop new applications using existing functionality. Some of these applications are detailed below. Links to these applications along with others can be found on our Web site at http://www.chembiogrid.org.

PubDock. We created a database of just under 1 million PubChem compounds docked into a small (but increasing) number of proteins taken from the Protein Data Bank using the OpenEye FRED docking program. This database is stored in PostgreSQL with the gNova CHORD cartridge and is exposed with a SOAP/WSDL interface. We have created several user interfaces to this database: a Web interface that allows the database to be searched and sorted by protein target, scoring function, and/or SMARTS including depiction of the docked complexes with Jmol and a Chimera-based interface that allows interactive visualization of compounds and the proteins they are docked to. The Chimera-based interface is shown in Figure 1.

PubChemSR. This is a desktop.NET application which allows search and retrieval of information in PubChem and export to Office tools such as Excel. The PubChemSR interface is illustrated in Figure 2.

R-Based Predictive Model Tools. We developed a number of Web services and associated Web applications for predicting properties and biological activities, using the R statistical package. These include toxicity and mutagenicity predictors using random forest models, *PKCell*, an application which calculates pharmacokinetic properties, and a generalized application which will allow an R-model to be built based on training data supplied in an Excel spreadsheet.

We have several other applications in development, and we hope that making our services publicly available will spur others to develop innovative applications.

USING WORKFLOW TOOLS

In addition to their use in application "mashups", Web services can be used in *workflow tools*. These tools permit the creation of new functionality by graphically linking together services and other applications and data sources into workflows (or pipelines). Workflow tools are increasingly being used in chemoinformatics, in particular Scitegic's Pipeline Pilot,[29] Inforsense KDE,[30] and more recently Knime.[31] This has resulted in some degree of exploitation of Web service workflows in the pharmaceutical industry, particularly for simplifying the task of using applications together. The use of Pipeline Pilot has been documented in a number of recent papers, which particularly focus on the use of the descriptors and Bayesian classifier supplied with the tool for virtual screening.[32−36] In addition, the Microsoft.NET framework also permits encoding of workflows of Web services.

We are using a variety of workflow tools at Indiana, including the open source Taverna[37] package. We have also implemented workflows in Pipeline Pilot, a local tool called XBaya, and using the Microsoft.NET workflow framework. We have developed several prototype workflows using our Web services and plan on making workflow files available on our chembiogrid.org Web site where possible.

FUTURE DEVELOPMENTS

An obvious next step would be the development of one or more Javascript APIs to some of the more generally useful services that can be made freely available (such as those

based on the CDK and R). This would enable these services to be used in standard Web pages, as a complement to the SOAP interfaces. We are working on such an API at the present time. We are also further developing prototype "mashup" applications as well as encouraging others to do so.

In the longer term, we are interested in the way that the Web service infrastructure can help to bring together the world of chemical information and chemical computation in conjunction with advanced query interfaces. We are also investigating the use of natural language interfaces for generating queries that involve both information retrieval and computation.

Finally, we are keen to encourage innovation in the development of more Web services and the development of applications and workflows that use the chemoinformatics infrastructure. We welcome the participation of others in the development of new services, mashups APIs, and query tools, and to that end we are actively participating in the Blue Oblelisk[38] movement. We are also committed to making all of our services publicly available except where we are bound by commercial licenses. The latest information can be found on our *chembiogrid.org* Web site.

ACKNOWLEDGMENT

This work was financially supported by the NIH through their Exploratory Centers for Cheminformatics Research funding and through a Microsoft Smart Clients for eScience grant. We would like to thank OpenEye, Digital Chemistry, and gNova for allowing us to use their programs. We would also like to thank Peter Corbett, Peter Murray-Rust, Gary Wiggins, T. J. O'Donnell, and Junguk Hur for helpful input into this work.

REFERENCES AND NOTES

(1) Google Maps API. http://www.google.com/apis/maps/ (accessed April 10, 2007).
(2) Bray, T.; Paoli, J.; Sperberg-McQueen, C. M.; Maler, E.; Yergeau, F. *Extensible Markup Language (XML) 1.0*, 4th ed.; W3C Recommendation. http://www.w3.org/TR/xml/ (accessed May 16, 2007).
(3) McGuinness, D. L.; van Harmelen, F. *OWL Web Ontology Language Overview*; W3C Recommendation. http://www.w3.org/TR/owl-features (accessed May 16, 2007).
(4) Gudgin, M.; Hadley, M.; Mendelsohn, N.; Moreau, J.; Nielsen, H. F.; Karmarkar, A.; Lafon, Y. *SOAP Version 1.2 Part 1: Messaging Framework*, 2nd ed.; W3C Recommendation. http://www.w3.org/TR/soap12-part1 (accessed May 16, 2007).
(5) Christensen, E.; Curbera, F.; Meredith, G.; Weerawarana, S. *Web Services Description Language (WSDL) 1.1*; W3C Recommendation. http://www.w3.org/TR/wsdl (accessed May 16, 2007).
(6) UDDI. http://www.uddi.org/ (accessed April 10, 2007).
(7) Berners-Lee, T.; Hendler, J.; Lassila, O. The Semantic Web. *Sci. Am.* **2001**, *284* (5), 34−43.
(8) Hendler, J.; Berners-Lee, T.; Miller, E. Integrating Applications on the Semantic Web. *J. Inst. Electr. Eng. Jpn.* **2002**, *122* (10), 676−680.
(9) Murray-Rust Group Web site. http://wwmm.ch.cam.ac.uk (accessed April 10, 2007).
(10) Murray-Rust, P.; Rzepa, H. S. Chemical Markup, XML, and the Worldwide Web. 1. Basic Principles. *J. Chem. Inf. Comput. Sci.* **1999**, *39* (6), 928−942.
(11) Murray-Rust, P.; Rzepa, H. S. Chemical Markup, XML, and the Worldwide Web. 2. Information Objects and the CMLDOM. *J. Chem. Inf. Comput. Sci.* **2001**, *41* (5), 1113−1123.
(12) Gkoutos, G. V.; Murray-Rust, P.; Rzepa, H. S.; Wright, M. Chemical Markup, XML, and the Worldwide Web. 3. Toward a Signed Semantic Chemical Web of Trust. *J. Chem. Inf. Comput. Sci.* **2001**, *41* (5), 1124−1130.
(13) Murray-Rust, P.; Rzepa, H. S. Chemical Markup, XML, and the Worldwide Web. 4. CML Schema. *J. Chem. Inf. Comput. Sci.* **2003**, *43* (3), 757−772.
(14) Murray-Rust, P.; Rzepa, H. S.; Williamson, M. J.; Willighagen, E. L. Chemical Markup, XML, and the Worldwide Web. 5. Applications of Chemical Metadata in RSS Aggregators. *J. Chem. Inf. Comput. Sci.* **2004**, *44* (2), 462−469.
(15) Holliday, G. L.; Murray-Rust, P.; Rzepa, H. S. Chemical Markup, XML, and the Worldwide Web. 6. CMLReact, an XML Vocabulary for Chemical Reactions. *J. Chem. Inf. Model.* **2006**, *46* (1), 145−157.
(16) Curcin, V.; Ghanem, M.; Guo, Y. Web services in the life sciences. *Drug Discovery Today* **2005**, *10* (12), 865−871.
(17) European Bioinformatics Institute. http://www.ebi.ac.uk (accessed April 10, 2007).
(18) Google Maps Mashup Resource. http://www.programmableweb.com/api/GoogleMaps/mashups (accessed April 10, 2007).
(19) AXIS libraries. http://ws.apache.org/axis/ (accessed April 10, 2007).
(20) Xfire. http://xfire.codehaus.org/ (accessed April 10, 2007).
(21) GNova CHORD. http://www.gnova.com (accessed April 10, 2007).
(22) OpenEye Software. http://www.eyesopen.com (accessed April 10, 2007).
(23) Digital Chemistry. http://www.digitalchemistry.co.uk (accessed April 10, 2007).
(24) World Wide Molecular Matrix. http://wwmm.ch.cam.ac.uk/ (accessed April 10, 2007).
(25) Townsend, J. A.; Adams, S. E.; Waudby, C. A.; de Souza, V. K.; Goodman, J. M.; Murray-Rust, P. Chemical Documents: Machine Understanding and Automated Information Extraction. *Org. Biomol. Chem.* **2004**, *2* (22), 3294−3300.
(26) Steinbeck, C.; Han, Y.; Kuhn, S.; Horlacher, O.; Luttmann, E.; Willighagen, E. The Chemistry Development Kit (CDK): An Open-Source Java Library for Chemo- and Bioinformatics. *J. Chem. Inf. Comput. Sci.* **2003**, *44* (6), 493−500.
(27) ToxTree. http://sourceforge.net/projects/toxtree (accessed April 10, 2007).
(28) Team, R. D. C. *A language and environmnet for statistical computing;* Foundation for Statistical Computing: Vienna, Austria, 2006.
(29) Scitegic Pipeline Pilot. http://www.scitegic.com (accessed April 10, 2007).
(30) Inforsense KDE. http://www.inforsense.com (accessed April 10, 2007).
(31) Knime. http://www.knime.org (accessed April 10, 2007).
(32) Klon, A. E.; Glick, M.; Davies, J. W. Combination of a Naive Bayes Classifier with Consensus Scoring Improves Enrichment of High Throughput Docking. *J. Med. Chem.* **2004**, *47* (18), 4356−4359.
(33) Klon, A. E.; Glick, M.; Davies, J. W. Application of Machine Learning to Improve the Results of High-Throughput Docking against the HIV-1 Protease. *J. Chem. Inf. Comput. Sci.* **2004**, *44* (6), 2216−2224.
(34) Klon, A. E.; Glick, M.; Thoma, M.; Acklin, P.; Davies, J. W. Finding More Needles in the Haystack: A Simple and Efficient Method for Improving High-Throughput Docking. *J. Med. Chem.* **2004**, *47* (11), 2743−2749.
(35) Xia, X.; Maliski, E. G.; Gallant, P.; Rogers, D. Classification of Kinase Inhibitors using a Bayesian Model. *J. Med. Chem.* **2004**, *47* (18), 4463−4470.
(36) Yoon, S.; Smellie, A.; Hartsough, D.; Filikov, A. Surrogate docking: structure-based virtual screening at high throughput speed. *J. Comput.-Aided Mol. Des.* **2005**, *19* (7), 483−497.
(37) Oinn, T.; Greenwood, M.; Addis, M.; Adlpdemir, M. N.; Ferris, J.; Glover, K.; Goble, C.; Goderis, A.; Hull, D.; Marvin, D.; Li, P.; Lord, P.; Pocock, M. R.; Senger, M.; Stevens, R.; Wipat, A.; Wroe, C. Taverna: lessons in creating a workflow environment for the life sciences. *Concurr. Comp.-Pract. E* **2005**, *18* (10), 1067−1100.
(38) Guha, R.; Howard, M. T.; Hutchinson, G. R.; Murray-Rust, P.; Rzepa, H. S.; Steinbeck, C.; Wegner, J.; Willighagen, E. L. The Blue Obelisk - Interoperability in Chemical Informatics. *J. Chem. Inf. Model.* **2006**, *46* (3), 991−998.

CI6004349