A Web / Grid Portal Implementation of BioSimGrid: A Biomolecular Simulation Database

Bing Wu[1,2,*], Matthew Dovey[2], Muan Hong Ng[4], Kaihsu Tai[1], Stuart Murdock[3,4], Hans Fangohr[4], Steven Johnston[4], Paul Jeffreys[2], Simon Cox[4], Jonathan W. Essex[3] and Mark S.P. Sansom[1]
[1]Department of Biochemistry, [2]e-Science Centre, University of Oxford,
[3]Department of Chemistry, [4]e-Science Centre, University of Southampton
*to whom correspondence should be addressed: bing@biop.ox.ac.uk

***Abstract**: The overall aim of the BioSimGrid project (www.biosimgrid.org) is to exploit the Grid infrastructure to enable comparative analysis of the distributed results of biomolecular simulations. In particular this paper presents the implementation of the current BioSimGrid Web Portal. The portal has a SOA (Service Oriented Architecture) framework built on the layer of OGSA (Open Grid Services Architecture) and OGSA-DAI (Open Grid Services Architecture Data Access and Integration) middleware. The PortalLib has been developed to allow RAD (Rapid Application Development) of portal applications. The portal also integrates PKI (Public Key Infrastructure) and supports two levels of distributed SSO (Single Sign On): Grid certificate-based SSO for high security, and user/password based SSO for maximal flexibility.*

Keywords: Biomolecular simulation, Open Grid Service Architecture, grid portal, BioSimGrid

Reviewed and accepted: 31 Mar. 2004

1. Background

Biomolecular simulations enable us to explore the conformational dynamics of complex molecules such as proteins, membranes and nucleic acids (Figure. 1). Molecular simulations with atom-level resolution have now entered the mainstream of biological research [1]. In particular, molecular dynamics (MD) is widely used to investigate nanosecond to microsecond dynamics for a wide range of biomolecules. Currently, a typical such simulation generates large amount of digital data.

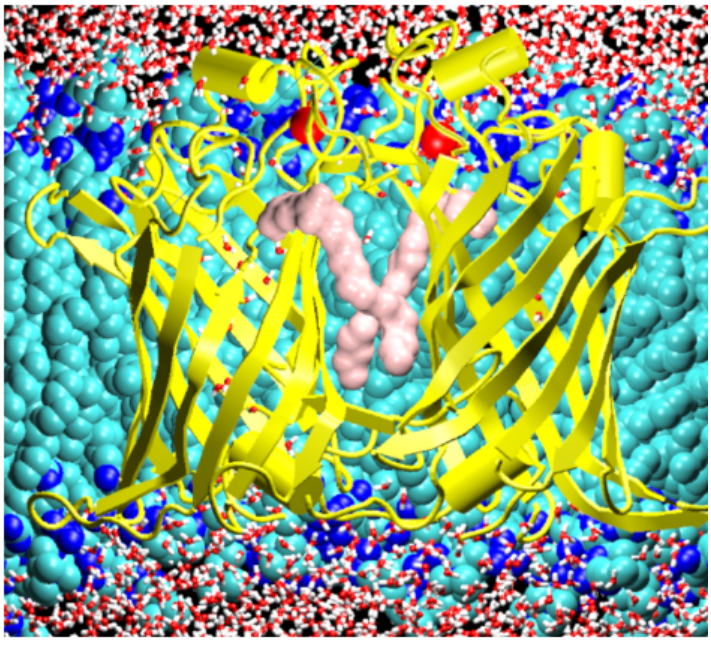

Figure 1. Comparison of protein simulations contribute to biomedical knowledge. Mouse acetylcholinesterase (left) and bacterial outer-membrane phospholipase A (OMPLA; right) are different in structure and biological environment (top) but similar in their active site (bottom) and catalytic function

MD has benefited considerably from improvements in computer technology. As computers become faster, biologists have become able to explore larger molecules for longer timescales. Currently, a typical simulation may have a system size of ~100,000 particles (atoms), and a nanosecond timescale simulation may require ~1,000,000 timesteps (i.e. iterations of integrating the equations of motion). Such a simulation would take a few weeks on between ~8 and ~64 processors (depending upon the efficiency of the simulation code and protocols employed) and could generate gigabytes of data for subsequent analysis and visualisation.

The status quo for the archiving of these data is far from optimal. Typically, data is archived in an ad hoc fashion at the level of individual laboratories. Furthermore, the reporting of the simulation metadata in journal articles is prone to omission of potentially important technical details. Consequently, even medium-scale comparisons between multiple simulations are not possible unless the simulations are performed within a single research group. This excludes simulation results from the domain of structural bioinformatics, where new information and knowledge is derived by comparisons between the results of individual research endeavors.

As protein structure determination becomes more automated, and as advances are made in structural bioinformatics [2] and computational biology, it will become increasingly important that biomolecular simulations do not exist as standalone analyses of single systems, but rather that they become embedded in a matrix of computational and experimental studies of proteins. However, at present there are considerable problems in comparing the results of multiple simulations [3], and in integrating simulation results with other (experimentally-derived) sources of data. This problem will become more pressing if simulation studies are to match up to post-genomic approaches such as high throughput protein crystallography. Furthermore, it will be essential to provide data retrieval and analysis tools that are accessible to a wide community of structural and cell biologists, not just to simulation specialists. It is in this context that the BioSimGrid [4] project is being developed.

2. Grid enabling database

The Grid [6, 7, 8] is a combination of network infrastructure and software framework delivering computing services based on distributed hardware and software resources. Using the power of the Grid the developers hope to solve the above problems, thus enabling them to take a comparative BioSimGrid approach to analysis of simulation data.

A major impediment to making various simulations available to biologists has been the absence of a database of simulation results. In an ideal world, all simulation data would be available to all interested parties. However, at present simulation data reside in the home laboratory and are not accessible to other research groups. One solution to this problem would be to deposit all simulation results in a centralised database, but in reality the vast quantities of data mean that rapid access would become impossible. The Grid provides the opportunity to draw together distributed collections of simulation data in disparate formats, whilst maintaining a centrally accessible federated database.

The BioSimGrid project will establish a formal database for biomolecular simulations within the UK, increasing collaboration via a distributed computing environment. There are three levels of data existing in the database (see Figure. 2):

- Raw data: generated by biomolecular simulations;
- 1st level metadata: describing the generic properties of raw simulation data;
- 2nd level metadata: describing the results of generic analyses of simulation data.

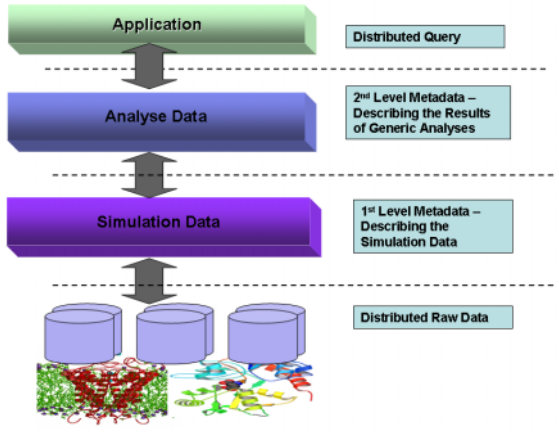

Figure 2. An overview of the BioSimGrid database

The core database has been implemented using IBM DB2 Database Version 8.1 Enterprise [5]. Current datasets contain 6 trajectories (two of which are trajectories of the nervous-system protein AChE; the others are of the bacterial outer-membrane protein OMPLA), 217 thousand atoms, 50 thousand frames at 1 picosecond apart, and about 1.7 billion coordinates. The data size is about 60 GB. These large amount of datasets need to be easily accessed by the community. To deliver this, we have designed and implemented a web portal.

A current direction of development focuses on the exploitation of Grid technologies such as OGSA-DAI [9] to enable interrogation of data through distributed queries. An application can send to BioSimGrid a query as if it were to be executed at one instance of a database, and BioSimGrid will process it to be distributed on several instances for execution.

3. Portal architectures

3.1. SOA - Service Oriented Architecture

The current methodology in developing distributed systems is Service Oriented Architecture (SOA), building upon methodologies such as Object Oriented programming, Components and Distributed Object Request Brokers. Within a SOA, systems are composed of multiple individual services located and maintained on different heterogeneous machines administered by different organizations. The key in SOA is that the component services should be loosely coupled to allow the orchestration of systems built up from component services, which should be robust against implementation changes in the underlying services [10]. To achieve this SOAs strive towards a number of goals:

1. The services should implement a small set of simple, ubiquitous and well known interfaces which only encode generic semantics.
2. The interfaces should deliver messages constrained by extensible schema for efficiency. This allows both services and consumers to work with well defined message structures, but allowing new versions of the services to be introduced without breaking existing systems.
3. The messages should be descriptive not instructive and the interfaces should not define system behaviour. This allows internals of a service can be viewed as a “black box”. You can send a service the details of the problem to be solved and preferences, but need not dictate how the service should solve the problem.
4. Service Oriented Architectures must have mechanisms for the discovery of services matching the consumers’ requirements.

There are a number of emergent technologies which can underpin SOA: REST Web Services; SOAP Web Services; and Grid Services.

3.2. REST, SOAP and Grid Services

Representational State Transfer (REST) works on the basic of “resources” which can be references by URIs [11]. A REST web service is limited to using HTTP interfaces (GET to obtain a representation of the resource; DELETE to remove a representation of a resource; POST to update or create a representation of a resource; PUT to create a representation of a resource). REST messages are in XML, constrained by schema definitions in XML Schema [12] or Relax NG[13].

SOAP Web Services use messages encapsulated in a structure defined by the SOAP specification [14]. This adds additional information in the form of headers for message routing scenarios and mechanisms for reporting errors using faults (a style similar to exceptions in various programming languages). SOAP Web Services use Web Service Description Language (WSDL) [15] to define both the structures (again using schema languages such as XML Schema) but also messaging semantics such as whether the message is initiated by the client or the server, and what messages can be used as a response to a particular message.

Grid Services are based on Web Services but provide additional semantics. In particular they add some object-oriented and REST concepts. The object-oriented concepts are the ability to inherit service definitions (portTypes in WSDL terminology) and add new messages using a multiple inheritance model and the ability to add properties (or service data elements) to Web Services. The REST concept introduced is that of creating a new representation of resource. In the Grid Services model this uses a factory model whereby a new instance of a Grid Service can be created by its corresponding factory Grid Service. Grid Services also offer an extensibility model whereby part of the structure of the message can be left undefined, but the allowed structures can be determined dynamically by querying the appropriate service data elements.

3.3. OGSI and OGSA infrastructure

As Grid Services offer an object-oriented inheritance model, the need for well known interfaces with an SOA can be met by defining base Grid Services which all Grid Services inherit. These foundation services form part of OGSI (Open Grid Services Infrastructure) and. OGSA (Open Grid Services Architecture), in addition to defining the extensions to WSDL needed for inheritance and service data elements and the semantics for the factory and extensibility models, also define a core set of foundation services and their behaviour. OGSA builds on top of OGSI to define various common service definitions for essential middleware components such as logging, account management, workflow management and data access [16].

The key middleware for BioSimGrid are the services defined within OGSA-DAI (Open Grid Services Architecture Data Access and Integration). This defines a set of services for accessing heterogeneous databases but with an abstraction layer so that the client can manage a table spread across multiple remote databases as if it was a table on a single local database [9].

3.4. Portal infrastructure

The key concept behind portals is content syndication. Portal channel producers publish raw content (with minimal presentation information), and it is the role of the portal to aggregate content and handle the presentation and rendering of content into a form suitable for the user. In this sense portals fit well into an SOA philosophy and provide a configurable and versatile user interface allowing the integration and management of loosely coupled services.

At present there are a number of emerging standards for communication between portal channels providers and consumers – JSR-168 defining Java interfaces [17] and WSRP (Web Services for Remote Portlets) defining a Web Service interface [18]. There is activity within the Global Grid Forum for an OGSA interface for portals. JST-168 and WSRP only finalized their specifications late 2003 and any OGSA based interface is still in early development and hence BioSimGrid had to develop its own portal library.

4. Portal implementation

4.1. SOA implementation

The current implementation of BioSimGrid Portal is based on multi-tier SOA architecture.

- GUI: HTTP(S)-based web client, provides user interaction with the system. The client can be either a standard web browser or web-based application. The use of the web interface eliminates development and maintenance of client software.
- Environment: This tier is a SOA front-end which handles communications between web applications and application servers. We have two environments here. The Web Portal Environment is a standard Apache/Tomcat based hosting environment to deliver web portal communications. The Python Hosting Environment is handling legacy Python applications.
- Application server: This tier is dedicated to delivering data analysis and data mining services through Grid/Web services. There are also supporting services such as monitoring, transaction, and

distributed query services. The protocols used here are XML/SOAP. The application servers also deal with OGSA-DAI Grid middleware to query the database.

- Database: IBM DB2 Database Version 8.1 Enterprise has been deployed as the core database. Data resources are distributed across collaborating sites. The OGSA-DAI Grid middleware enables the access of these resources transparently in a format of virtual machine. The database of an individual site can be clustered to achieve better performance and reliability.

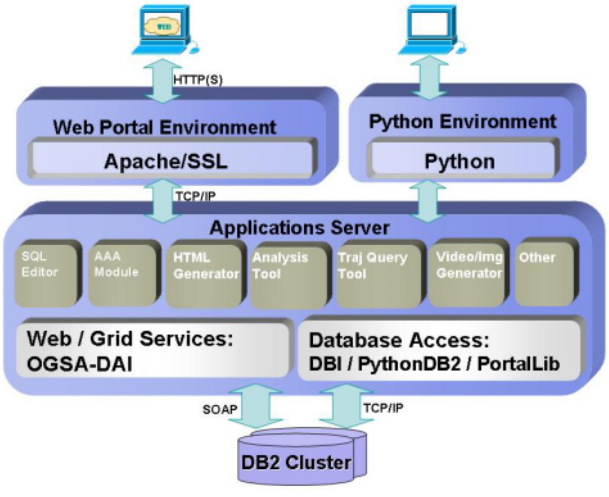

Figure 3. BioSimGrid Portal implementation

4.2 Dealing with simulation data

We have selected an initial application that both tests our methodology and also allows us to explore an important biological question. We are currently using BioSimGrid to compare simulations of two enzymes: (i) acetylcholinesterase (AChE), a key enzyme of the nervous system [19]; and (ii) outer-membrane phospholipase A (OMPLA), a bacterial enzyme involved in pathogenesis [20]. Structural data show that these two enzymes have similar active sites (a triad of amino acid sidechains involved in their catalytic mechanisms). Otherwise, the structures of the two proteins are not closely related.

We are analysing simulations to compare the patterns of catalytic sidechain dynamics in these two distantly related enzymes, so as to understand the relationship between sidechain mobility and catalytic mechanism. The AChE simulations were performed at University of California, San Diego; the OMPLA simulations in Oxford. Normally, such data would never “meet” and would reside in the separate laboratories. Furthermore, the simulations were run using different programs and protocols and the data are in very different formats. Rather than re-run one (or both) of the simulations (which would consume many days of costly supercomputer time), we are using BioSimGrid to make the comparison. Thus this apparently simple test case enables us to test many aspects of the underlying methodology.

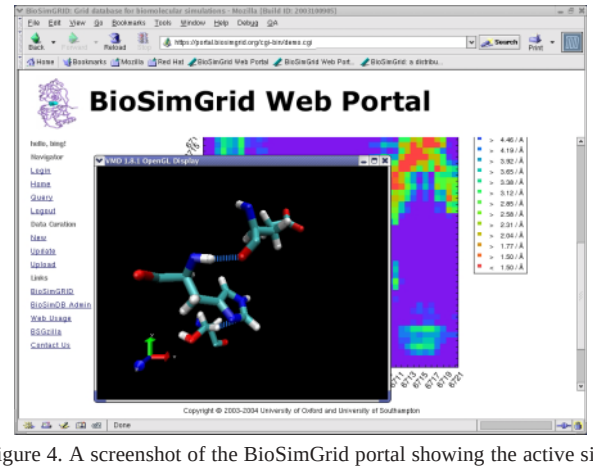

Figure 4. A screenshot of the BioSimGrid portal showing the active site for AChE

4.3. PortalLib – delivering the web front-end

A web portal implementation involves a large number of web pages, which are based on HTML code. Among these, a certain amount of the web pages have to be generated dynamically based on user requests. There are basically two ways of implementation: client-side scripting and server-side scripting. In the BioSimGrid Portal implementation, we have both implementations. JavaScript is currently used on the client-side as it can run on most web browsers, while Perl and Python are used on the server-side. We developed PortalLib to provide a common interface for generating dynamic web pages on the fly. PortalLib is a Perl library which consists of four main classes to use for rapid portal development:

- HtmlFile - interfaces with static HTML template files to generate dynamic web pages,
- HtmlComponent -a standard HTML component to generate web GUI,
- Auth_lib – provides common Authentication and Authorisation interfaces,
- DB_lib – provides a database common interface to DBI compatible data sources.

The code in Table 1 is an example of using PortalLib to produce a trajectory refinement form based on previous user inputs and database query results. When running ‘refine.cgi’, the result data is dynamically bound to the template HTML file called ‘refine.html’. For a typical portal page, we place labels, text fields, buttons and scrolling lists in the template page and map them to the attributes in the hash list of ‘%options’. Then the attribute data will be automatically passed to the next Web page via HTTP session. Thus by using PortalLib, we have achieved rapid and efficient development of server-side web pages.

```
#!/usr/bin/perl -w

use PortalLib;
use strict;
use CGI;

my $cgi = new CGI;
my %options = ( $cgi->Vars,);

# process template HTML
my $template = ‘refine.html’;

my $html = HtmlFile->new(
  File => $template,
  %options,
  );

$html->Draw();
```

Table 1. Example code: refine.cgi

JavaScript can also be dynamically passed to an HtmlFile object in the JavaScript property or embedded in a template file as usual HTML code. The generated form is shown in Figure 5. The trajectory list is pulled dynamically from the database and bound to the web front-end below.

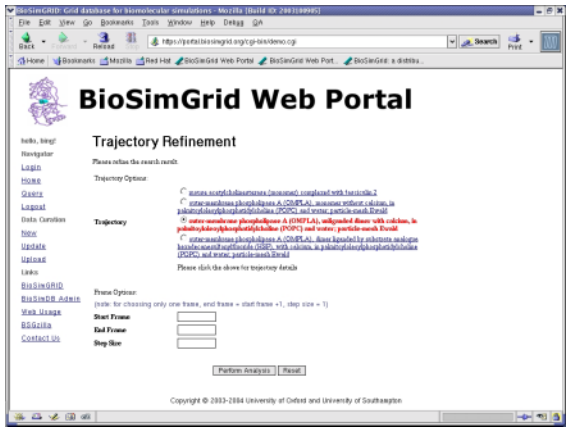

Figure 5. Trajectory refinement form

5. BioSimGrid security

Given the current distributed Grid implementation, security is a critical element of the project. We integrated various mechanisms to achieve a secure distributed environment. We have implemented PKI (Public Key Infrastructure) and X.509 Digital Certificate [21] based authentication. All the transactions are based on HTTPS secure channels. The BioSimGrid network has a dual firewall protection. Two levels of authentication enable legacy user/password based authentication. The authorisation is based on a distributed database and implements SSO (Single Sign On).

5.1. PKI and Grid certificates

PKI is a system of public key encryption using digital certificates from a CA (Certificate Authority) that verifies and authenticates the validity of each party involved in an electronic transaction. PKI uses an asymmetric key based algorithm: a private key is used to encrypt data and a public key can decrypt data encrypted with the private key. A certificate contains information referring to a public key which has been digitally signed by a CA. The information normally found in a certificate conforms to the ITU (IETF) standard X.509 v3 [22].

A Grid certificate is an X.509 certificate. We have two kinds of Grid certificates used in the portal: user certificates for user identities and host certificates for servers. Once you have a valid user certificate, you can use it to access the web portal [24]. A host certificate has similar format as the user certificate except its DN has a hostname instead of the user name in 'CN'.

subject=/C=UK/O=eScience/OU=Oxford/L=OeSC/**CN=portal/portal.biosimgrid.org**/emailAddress=bing@biop.ox.ac.uk

The host certificate can be configured to be used by the web server supporting SSL transactions. We use the Apache server in the portal environment. The above host certificate is converted to an Apache-friendly key pair and loaded into the web server [25]. The same key pair can also be used as a host certificate in the Globus Toolkit [6].

5.2. SSO (Single Sign-On) and AAA (Authentication Authorisation Accounting)

SSO is designed to provide a foundation that gives users role-based access to multiple Web applications from a single, secure point of contact. This simplifies the user AA (Authentication Authorisation): the user only needs to remember one user id/password. There are various implementations of SSO technologies, i.e. 3rd party based SSO and centralised SSO.

In the BioSimGrid project, we have a distributed Grid environment and need to deal with two levels of authentication for high security and easy accessibility. To achieve this, we use SSO based on a distributed database. All the user accounts and corresponding AAA information are stored in the database distributed across the network. This enables a user sign on at any BioSimGrid site to access authorised resources and perform authorised transactions. The accounting information of the user access will be stored locally and distributed over the network.

As in Figure. 6, two levels of authentication infrastructures have been implemented in the system. The first level is based on digital certificate. A Grid certificate-based authentication mechanism has been integrated across the system. When a user signs on to access specific BioSimGrid services, the subject of his/her X.509 personal digital certificate is passed to the SSO Security Check module, which then calls the AAA module to verify the certificate against the one stored in the accounts database. Only if the security check is successful will the user have the access to the services. The second level is a user/password based authentication, which is designed for those who have no digital certificates installed in their client machines. This level of authentication enables web access of the portal via a public PC from anywhere in the world.

Once the user has been granted access, the AAA module will store the user credentials in the database and generate a unique access token for this session. The token has a limit life time to enhance the security. We are also investigating the integration of user credential delegation using MyProxy [26]. All transactions are logged in the accounts database, which will then be distributed across the BioSimGrid sites for maximal efficiency and high availability.

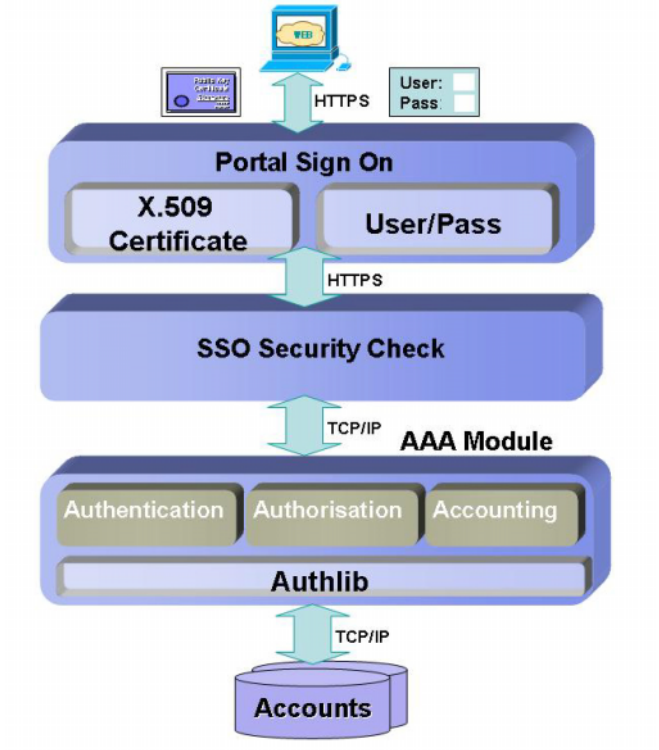

Figure 6. BioSimGrid SSO implementation

6. Conclusions

The BioSimGrid project is still in its infancy. However, the AChE vs. OMPLA comparison provides a microcosm of the many comparisons that will become possible once BioSimGrid is fully operational. Biomolecular simulation groups will be able to deposit their simulation data for wide-ranging comparative analyses that so far have been impossible. This will help to propel biomolecular simulation studies into the post-genomic era.

The OGSA and OGSA-DAI architectures underlying BioSimGrid are still in the early stage of development. Therefore we have both legacy programs and new web services co-existing in the current portal environment. Once Web / Grid services are stable, it would be worthwhile to migrate all legacy programs to web services under the proposed SOA architecture.

We are developing methods for data deposition, data exchange and for quality control of simulation data. This will enable us to run initial comparative analyses on multiple simulations (e.g. of membrane proteins) in order to evaluate the current prototype in real world applications.

Acknowledgements

Many thanks to our BioSimGrid partners (L. Caves, C. Laughton, D. Moss and O. Smart) for their input to this project. BioSimGrid is funded by BBSRC and DTI. Our thanks to all of our colleagues in the Oxford and Southampton simulation labs, to Ivaylo Kostadinov in the Oxford e-Science Center, and to the Southampton Regional e-Science Centre for their encouragement, advice and hard work. Our thanks also to Marc Baaden for providing the OMPLA trajectories and figure1.

References

1. Karplus, M.J. and McCammon, J.A. (2002). *Nature Structural Biology*., 9, 646-652.

2. Bourne, P.E. and Weissig, H. (2003). Structural Bioinformatics, Wiley-Liss, Hoboken.

3. Pang, A., Arinaminpathy, Y., Sansom, M.S.P. and Biggin, P.C. (2003). *FEBS Lett.550*, 168-174.

4. Bing Wu, Kaihsu Tai, et al. (2003). BioSimGrid: A Distributed Database for Biomolecular Simulations. Simon J Cox (editor), 5. 5. Proceedings of UK e-Science All Hands Meeting 2003. EPSRC, ISBN 1-904425-11-9.

5. http://www-3.ibm.com/software/data/db2/udb/

6. http://www.globus.org

7. Berman, F., Fox, G. and Hey, T., Eds. (2003). Grid Computing: Making the Global Infrastructure a Reality, Wiley.

8. Foster, I. and Kesselman, C., Eds. (1999). The GRID: Blueprint for a New Computing, Morgan-Kaufmann.

9. http://www.ogsa-dai.org

10.Hao He (2003). What is Service-Oriented Architecture, http://webservices.xml.com/pub/a/ws/2003/09/30/soa.html

11.http://www1.ics.uci.edu/%7Efielding/pubs/dissertation/rest_arch_style.htm

12. http://www.w3.org/XML/Schema

13. http://www.oasis-open.org/committees/tc_home.php?w_abbrev=relax-ng

14. http://www.w3.org/2000/xp/Group/

15. http://www.w3.org/2002/ws/desc/

16. Foster, I., Kesselman, C., Nick, J. and Tuecke, S. (2002). The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration, Global Grid Forum.

17.http://www.jcp.org/aboutJava/communityprocess/reiew-/jsr168/

18. http://www.oasis-open.org/committees/tc_home.php?w-_abbrev=wsrp

19. Kaihsu Tai, Tongye Shen, Richard H. Henchman, Yves Bourne, Pascale Marchot, J. Andrew McCammon (2002). Mechanism of acetylcholinesterase inhibition by fasciculin: a 5-ns molecular dynamics simulation. *Journal of the American Chemical Society* 124:6153-6161.

20. Baaden M, Meier C, Sansom MSP (2003). A molecular dynamics investigation of mono- and dimeric states of the outer membrane enzyme OMPLA. *Journal of Molecular Biology* 31:177-189.

21. Tuecke, S., Engert, D., Foster, I., Thompson, M., Pearlman, L. and Kesselman, C. (2001). Internet X.509 Public Key Infrastructure ProxyCertificate Profile, IETF.

22. ftp://ftp.isi.edu/in-notes/rfc2459.txt

23. http://www.grid-support.ac.uk/ca/

24. Bing Wu (2003). User Certificate Installation Guide, http://www.biosimgrid.org/docs/2003/NOTES/user_certificates.pdf

25. Bing Wu (2003). Digital Certificate Installation Guide, http://www.biosimgrid.org/docs/2003/NOTES/certificates.pdf

26. http://www.ncsa.uiuc.edu/Divisions/ACES/MyProxy/