CONCURRENCY AND COMPUTATION: PRACTICE AND EXPERIENCE
Concurrency Computat.: Pract. Exper. 2008; **20**:1095–1111
Published online 9 January 2008 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/cpe.1285

A service-oriented Grid environment for integration of distributed kidney models and resources

Xingchen Chu[1], Andrew Lonie[2], Peter Harris[3], S. Randall Thomas[4] and Rajkumar Buyya[1,*,†]

[1]*Grid Computing and Distributed Systems Lab, Department of Computer Science and Software Engineering, The University of Melbourne, Melbourne, Vic., Australia*
[2]*Department of Information Systems, The University of Melbourne, Melbourne, Vic., Australia*
[3]*Faculty Information Technology (IT) Unit, Faculty of Medicine, Dentistry and Health Sciences, The University of Melbourne, Melbourne, Vic., Australia*
[4]*IBISC (Informatiques, Biologie Intégrée et Systèmes Complexes) FRE 2873, CNRS/Université d'Evry-Val d'Essonnes, 91000 Evry, France*

SUMMARY

In this paper, we present a Grid computing platform that provides experimental scientists and analysts with access to computational simulations and knowledge databases hosted in separate laboratories around the world involved with human and animal kidney research. No single laboratory can develop these resources in isolation, and the community of users should no longer need to be dependent upon the specific programming environment in which applications have been developed. This work aims at exploiting the power of high-bandwidth communications networks for collaborative research and for shared access to knowledge resources. This platform is developed within a specialist community of renal scientists but will be transferable to any other field of research requiring interaction between published literature and databases, theoretical models and simulations and the formulation of effective experimental design. Copyright © 2008 John Wiley & Sons, Ltd.

Received 27 February 2007; Revised 15 June 2007; Accepted 1 October 2007

KEY WORDS: kidney modeling; Grid computing; Grid portal; Gridbus resource broker; Web service resource framework; visualization

*Correspondence to: Rajkumar Buyya, Grid Computing and Distributed Systems Lab, Department of Computer Science and Software Engineering, The University of Melbourne, Melbourne, Vic., Australia.
†E-mail: rbuyya@unimelb.edu.au, raj@csse.unimelb.edu.au

Contract/grant sponsor: ARC (Australia Research Council)
Contract/grant sponsor: DEST (Department of Education, Science and Training)

Copyright © 2008 John Wiley & Sons, Ltd.

1. INTRODUCTION

Mathematical modeling has been used successfully in analyzing the huge volume of biomedical data resulting from such ventures as cardiac modeling. In particular, the approach has been very successful in modeling heart physiology, and mathematical modeling has always played a key role in research on kidney physiology. A large number of mathematical models and resources describing aspects of kidney physiology exist, in different formats and simulation environments around the world; however, they are generally proprietary and incompatible with each other, making access and integration of models and resources extremely difficult and time consuming. A Grid computing-based approach to integrating and representing distributed renal models and resources will provide researchers with ready access to a collection of interactive models and resources that minimize the requirement for specialist programming or computational modeling expertise on the part of the experimenter. A variety of legacy kidney models [1–10] exist in the research community, representing many different implementation techniques (different modes of interaction—command line, graphical interface, command file driven) and they run on several different platforms. It is very difficult to (i) accelerate the development process for each model since they are distributed among diverse computing platforms and geographical areas, and (ii) manage the community of users and the information provided by the models. Grid computing [11] not only provides powerful computation facilities as a means for researchers to do existing research faster but also promises them a number of other facilities such as working in collaborative environments, reducing costs and gaining access to an increased number of resources and instruments, allowing for more advanced research to be carried out by a wider community of users. Furthermore, many of the complexities arising with Grid computing have been isolated and solved by technologies including Grid middleware [12], resource schedulers [13], virtual organization [14], data management [15] and resource management, and information services. Advanced integration of these technologies generates the basic services of community infrastructure for life science and other e-Science/e-Research areas, as shown in Figure 1.

In this paper, we describe the development and deployment of an interactive Web-based portal for quantitative renal physiology modeling at the international level, with the primary aim of providing general access to a repository of legacy renal models at all scales and to the published measurements that underpin the parameter values needed for the development of such models. Specifically, we are able to provide an infrastructure with the following features:

- Additional functionality through Web-based access to published quantitative measurements on all aspects of renal physiology and anatomy (QKDB) [16] and to the legacy of hypothesis-based mathematical models based on these data.
- A collaborative environment to encourage and assist the entry of new workers into modeling, particularly those with less well-developed skills and expertise in programming and computational biology.

The infrastructure includes an interactive Web interface to a collection of distributed legacy models at all levels of kidney physiology, with, for each model, documentation, physiological context, easily interpreted output, a statement of model limitations, interactive exploration and user-customization of selected parameter values. A 3D kidney virtualization graphical user interface is used to facilitate interaction with the resource. This project is part of the

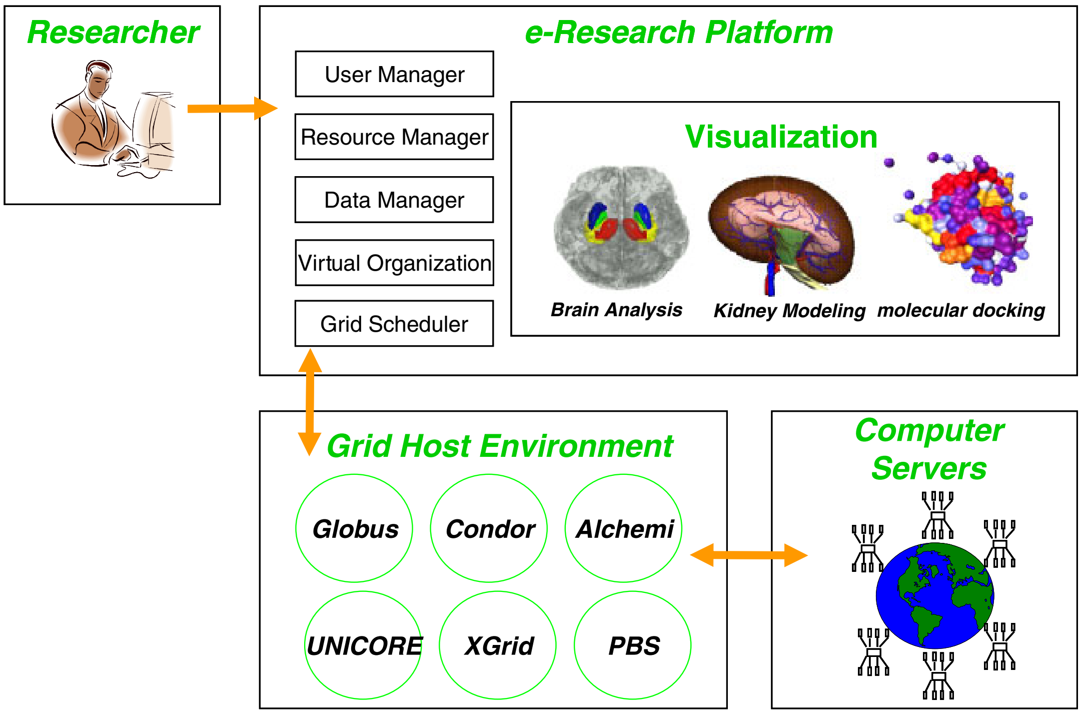


Figure 1. Typical e-Science/e-Research platform.

larger IUPS Physiome Project, an internationally collaborative open-source project to provide a public domain framework for computational physiology, including the development of modeling standards, computational tools and Web-accessible databases of models of structure and function at all spatial scales, namely interactions, protein pathways, integrative cell function, tissue and whole organ structure–function relationships, and finally the integrative function of the whole organism [17].

The remainder of the paper is organized as follows. In Section 2, we describe related work in the e-Science and life science areas. The design issues of our infrastructure are introduced in Section 3 and in Section 4 we present details of the design and implementation of our KidneyGrid. The evaluation of one scenario we have developed, which integrates a legacy model of renal medulla called kidney simulation (KSim) under our environment, is described in Section 5. Finally, a brief conclusion and plans for future work are presented in Section 6.

2. RELATED WORK

Many efforts have been made to develop e-Science platforms and infrastructure in the Grid environment. A portal for Grid-enabled physics, in particular astrophysics and high-energy particle physics, was proposed and implemented in [18]. It was able to host user-level services such as simulation, or handling analysis of job configuration, job submission and monitoring. The Australian BioGrid Portal [19] was another portal-based solution enabling bio-scientists to perform drug-lead exploration in an efficient and inexpensive manner on national and international computing Grids [20]. It provides a complete molecular docking e-Science platform for the use of docking software, from user management to experiment composition, execution and monitoring over Grid resources, and finally visualization of simulation results. A Grid platform for distributed analysis of neuroimaging data was introduced in [21], which was able to compose and formulate magnetic resonance imaging (MRI) data analysis as a parameter sweep application and perform the experiments on the Grid resources. It substantially reduced the total time needed to analyze MRI data compared with previous solutions. As reported in [22], a chemistry portal was developed for Grid-enabled molecular science and, similar to the BioGrid [19], also adopted a portal-base solution for executing experiments over Grid resources for quantum chemistry code. Importantly, the solution [22], unlike [18,19,21], did not utilize a user-level Grid middleware such as a resource scheduler [13]. The advantage of utilizing the user-level resource scheduler is that it decouples the system and underlying Grid middleware such as Globus [12] and makes the system applicable for other Grid resources without the support of Globus. In this work, we take an approach similar to that described in [18,19,21,23], but focus heavily on the utilizing Web services-based technologies to enhance interoperability with other resources/ platforms.

3. SYSTEM OVERVIEW

3.1. Architecture

We aim to develop a life science infrastructure and collaborative computing environment that enables: (a) formation of a distributed community of developers of kidney models, resources and users as a virtual organization; (b) integration of distributed kidney models provided by different development communities; and (c) Web portal-based interaction between various models allowing aggregation and advanced visualization. A service-oriented Grid-based architecture for realization of this life science environment for distributed kidney modeling is shown in Figure 2. The entire infrastructure helps in overcoming many of the limitations we faced in our early proof of concept of the Web-based portal for kidney models, in which all interactions with remote models were manual and hard-coded; communication was mediated through specific protocol calls (e.g. scripted telnet sessions), or results files were simply uploaded to the portal server using HTTP or FTP, and new resources and users were added manually. To this end, the proposed development of a Grid-based infrastructure for managing the distributed components enables a vastly simplified interaction mechanism for both model 'suppliers' and users of the system.

High-level interactions between various entities, components, resources, and human participants and the channels of control (shown in Figure 2) are described below:

1. User initiates session with portal by selecting modeling resources required and 'planning' experiment with the resources.
2. User obtains authorization through VO and authorization service.
3. Portal contacts Grid broker to request resources (remote/local kidney database or remote simulations).

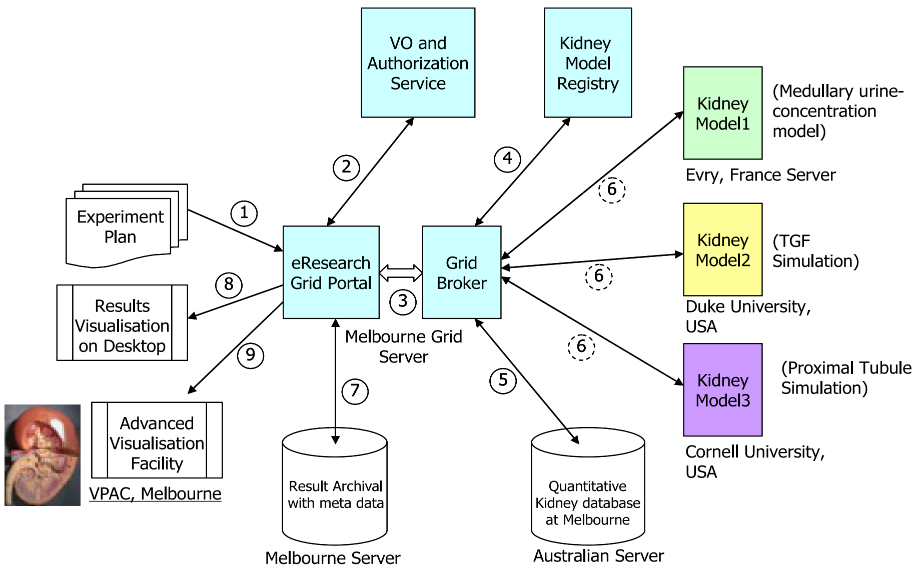


Figure 2. A Grid-based e-Research architecture for integration of distributed kidney models.

4. Grid broker queries the registry to identify resources for certain kidney model. Registry returns list of resources required by the broker to schedule the simulations.
5. Broker opens communication channel to remote resource and sets up experiment information. Note that resources may be locally based (e.g. in the local mirror of the kidney database) or remote.
6. More commonly, simulations will be hosted on remote sites, running on proprietary systems. The Grid broker will manage communications with the remote resource through a combination of communications and/or implementation of service-based wrappers at the remote resource. A basic assumption of our system is that as little as possible should be required to be implemented on the remote resource; hence, communication brokering will most likely be tailored to individual model resources with most data processing of result sets occurring at the portal.
7. User session state, including current state of experiment, is saved to local database.
8. Results are visualized on the users' desktop through the Web browser.
9. Alternatively, for complex data sets we will make use of external visualization resources, in particular the VPAC Advanced Visualization Facility in Melbourne.

3.2. Issues to be addressed

In the long term, we aim to develop a platform for 'plug and play' modeling, with a standard basis of representation for all models (preferably an XML-based open modeling markup language such as CellML [24]) and with similarly standardized data representation. We have made initial proof-of-concept progress toward this goal by developing a CellML-based implementation of a published model (although the input and output data formats are still 'proprietary' to some extent) and rendering the results in an open standards-based 3D interface [25]. The implementation is based on the model described in [1]; currently, we have two cell types working with advection–diffusion spatial modeling and have demonstrated agreement of the published numerical output [25]. Outputs of this model, including solute concentrations along the length of the distal tubule, are rendered on a browser-based 3D nephron representation derived from real microCT data, which we intend to develop into the primary graphical interface for interaction with the KidneyGrid resources data representation.

It is important to include sufficient material for the site to be interesting and useful once it is opened to the general community. The models we will implement in the initial core collection cover a wide range of scales and represent some of the classic studies in the field. The list includes:

- distal tubule reabsorption [1];
- proximal tubule reabsorption [2,3];
- tubulo-glomerular Feedback [4,5];
- glomerular filtration [6];
- medullary models of the urine-concentrating mechanism:
    - simple central core model [7,8];
    - multi-individual-nephron flat model [9];
    - shunted-multi-nephron flat model [10].

It should be noted that the listed legacy kidney models and resources represent many different implementation techniques (different modes of interaction—command line, graphical interface, command file driven) and they run on several different platforms. There are currently no standards for data representation, interaction or communication.

The issues of standards and interoperability could be addressed by recoding all the models in a common format as discussed above, but this long-term and resource-intensive solution is not feasible without direct and significant input by each of the model developers. A more practical approach is to develop interaction 'wrappers' around the current model implementations and within the central portal convert the output so that diverse result sets can be rendered on a common interface. This preferred approach requires infrastructure 'middleware' to manage the communication channels, data transfer channels, user session state and user authentication within the context of a portal.

The basic principle is to create a Web page that presents the user (i.e. the visitor to the Web site; also called the 'client') with an annotated list or a visual interface for selection of one of the available models. The user is then presented with a more detailed description of the chosen model, including literature references, and is given the opportunity to either look at the results of some pre-configured simulations to 'get the feel' of the model, or to alter particular parameter values (within a constrained range representing physiologically reasonable values) in order to run the model as required. Upon submitting the new parameter values, the user will be served (after a wait that depends on the execution time of the model calculations) with the results and may choose among several styles of presentation, such as a simple table, graphical output, or in some cases more visually meaningful presentations, or may request to download the results as a table for post-treatment in a spreadsheet or graphing program of their own.

4. DESIGN AND IMPLEMENTATION

The primary design principle of our KidneyGrid is to maximize the reuse of existing software components, tools, services and frameworks provided by the open-source community for generic tasks, while concentrating our efforts on the integration of those technologies as well as providing the service wrapper for legacy kidney models. Table I provides details on key technologies used in the development of our KidneyGrid environment. Thus, we are able to leverage various existing Grid technologies and build on our experience in the development of Grid-based platforms for e-Science applications in the life sciences area.

4.1. Portal development

Gridsphere was chosen as the portal development framework because (i) it is an open-source and JSR 168 compliant, which means we could easily reuse the components in other portal containers supporting this standard; (ii) it supports the Java Server Facers (JSF) standard, which could potentially be used within a visual page editor tool to accelerate the development of Web pages; (iii) it contains common portal components such as user management and localization support, which we could reuse directly; and (iv) it is mature and widely used by other Grid portal-related projects.

In our portal, we have reused the user management and authentication module provided by the Gridsphere. We have developed a project management module that allows multiple users to

Table I. Key technologies chosen.

Component	Technology used	Comments
Portal	Gridsphere framework [26]	Reusable Web components. JSR 168 portlet standard compliant. Open source and widely used for Grid portal development
Virtual organization (user authorization)	MyProxy [27] VOMS [14]	VO-based authorization and application scheduling
Grid resource broker	Gridbus broker V3.0 [13]	Provide ability to support Web service resource framework (WSRF), various scheduling algorithms and resources such as Globus and Alchemi
Service wrapper for legacy kidney model	Web service resource framework (WSRF) [28]	WSRF provides stateful Web service which is able to be deployed on the Globus toolkit 4 (gt-4) container
Visualization	VPAC Advanced Visualization Facility in Melbourne	Provide Java applet that can be easily integrated into Web portals
Core Grid Middleware	Globus Toolkit 4, Alchemi, UNICORE	Provide supports for integration of Grid resources accessible through different low-level Grid middleware

manipulate their own projects and kidney model simulations. The resource management module has also been developed for administrator to manage the available Grid resources. Furthermore, in order to include the new kidney models that can be executed with the portal, a kidney model registration module has been developed that allows the administrators to add new kidney models into the portal environment. Users can also manage their credentials through the credential management module we have developed. For each kidney model, we have developed different experimental preparation modules, that are specific to that model with certain parameter settings. It is highly configurable through the JSF XML configuration file to add the preparation modules for new introduced kidney models in the portal. For invoking the remote kidney models, we have used the portlet-based job execution management on Grid resources provided by the Gridbus broker. The execution monitor module that has been used in this portal is derived from the Gridbus broker execution monitor portlet and we extended the monitor portlet using the JSF technology. Other modules in the portal include the visualization module, which is also designed to be highly configurable for various kidney models, and the repository module, which allows users to store their experimental results into the local repository for their future comparison with the latest experiments.

4.2. Legacy application integration

The most important feature of our KidneyGrid is the integration with legacy kidney models developed by different programming languages and platforms. The simplest approach is to allow the clients to explicitly upload the executable command, program arguments and data files to the resources they are using. The user takes full control of the process to run the application on the server and is effectively responsible for using the middleware to Grid-enable the application. However, it assumes the client who runs the application has an existing account on the remote resources and it also substantially increases the dependencies between the client and the service implementations. Our approach is to provide both language and platform neutral and interoperable service wrappers for each model with those wrappers, being responsible for the inter-communication between our infrastructure and the legacy models. The application that performs the service is hidden from the client and there is no need for the user to have an account on the service provide site. Furthermore, the wrapper should be able to be deployed inside a Grid environment such as Globus [12] and Alchemi [29]. The wrapper should also be capable of storing the states of the input, output and execution information as some of the model simulations may continue over long time periods.

The Web service resource framework (WSRF) [28] provides a service-oriented architecture (SOA) solution to keep state as separate resources called WS-Resource, which do not require the Web services to keep track of the resource identifier explicitly by passing it as a parameter each time. The complete separation of Web services and state information as WS-Resources retains the relatively similar scalability of stateful Web services compared with plain stateless Web services, and fulfills the requirement for keeping state information of Grid applications. WSRF relies on a set of specifications, including WS-ResourceProperties, WS-ResourceLifetime, WS-ServiceGroup, WS-BaseFaults, WS-Notification and WS-Addressing.

The wrapper service for the legacy kidney models utilizes WSRF, implementing one kidney WS-Resource for each model representing the state information such as input and output parameters, and execution outcomes. Since a given simulation of the kidney modeling may not respond immediately, we also utilize the notification mechanism in WSRF provided by WS-Notification to notify the client once the simulation has finished and to collect the result. For each wrapper service, we implement a factory service that is responsible for creating the kidney resources and returning the handler of the actual service that performs the modeling simulation. As mentioned, the legacy kidney models use various programming languages. We therefore implement an executor written in Java for each model, which simply creates a runtime process for the modeling program and passes the parameters or composes the input files at runtime. The wrapper service actually utilizes the executor and passes the required parameters.

4.3. Grid resource broker

Consider the scenarios where some of the legacy models are only compatible with the UNIX environment and some are only executable under Windows. In these cases, we require different Grid host environments, for example, Globus [12] for UNIX and Alchemi [29] for Windows and it is very important to utilize a user-level middleware to decouple the management work such as job composition, scheduling, monitoring and result collection from the underlying Grid middleware. Gridbus Broker [13] is a user-level meta-scheduler that mediates access to distributed resources and provides facilities for all the aspects of the application, credential and service management. The broker includes appropriate plug-ins for most user systems and provides such a seamless way for clients to schedule their applications on heterogeneous resources such that they do not need to be concerned with underlying systems. An economic-based administration of modeling is also supported by the broker such that users can specify budget and deadline for their application. In addition, various scheduling algorithms have been provided to fulfill the quality of service (QoS) requirements. We adopt version 3.0 of the Gridbus Broker, which works with remote services that are compatible with WSRF-based Web services. It supports various commonly used low-level middleware such as Globus 2, Globus 4, Alchemi, UNICORE, and SSH-based connection to PBS and SGE. The broker services can be invoked as Web services via SOAP, meaning that broker can be deployed separately on other servers instead of the same server on which our portal is running. The system scalability can be substantially increased by running the broker as a WSRF service on another server or multiple servers and the reliability of the portal is also enhanced as the failure rate of the broker is potentially much higher compared with the portal itself.

4.4. VO-based authorization and scheduling

As the kidney models may come from different universities, research institutions or industries, a virtual organization (VO), which is an abstract entity grouping users, institutions and resources in a same administrative domain, could be assigned to each kidney model. By applying VO concepts, we could have multiple VOs that are mapped to various kidney models. As the user may belong to several different VOs or have multiple roles in the same VO, authorization based on the user information in each VO becomes very important. We have addressed this requirement by implementing a module including a VOMS client and a VO policy engine through which our platform can query an existing VO system (VOMS [14]) and extract policy information inside each VO to authorise the client and then schedule the application according to the policy set by our resource broker.

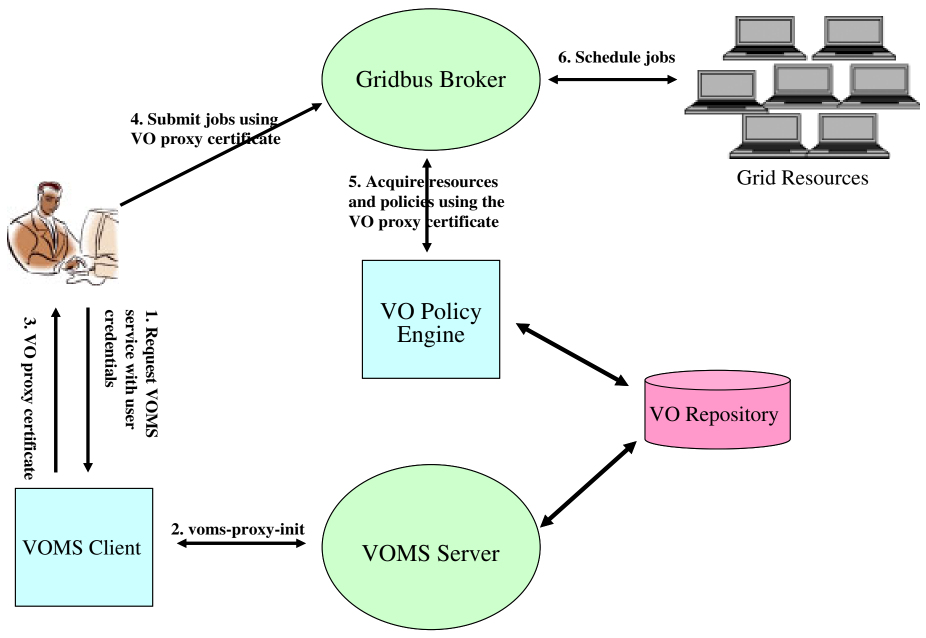

Gridbus Broker
6. Schedule jobs
Grid Resources
4. Submit jobs using VO proxy certificate
5. Acquire resources and policies using the VO proxy certificate
VO Policy Engine
VO Repository
1. Request VOMS service with user credentials
3. VO proxy certificate
VOMS Client
2. voms-proxy-init
VOMS Server

Figure 3. VO-based authorization and scheduling components architecture.

As shown in Figure 3, the VOMS client is responsible for contacting the VOMS server to issue the VOMS proxy certificate to the end user. We assume that the end users only have their normal X509 proxy certificates and in order to obtain the VOMS proxy certificate they need to contact the VOMS client and provide their proxy certificates to it. The VOMS client will then ask the VOMS server for the VOMS proxy certificate with the user's proxy certificate. After the execution of voms-proxy-init, the VOMS proxy certificate, which contains the VO information, will be issued by the VOMS server to the clients. As soon as the clients receive their new certificates, they can submit jobs to the remote resources via the resource broker. The broker has been extended to validate the new certificates and parse the VO information including the VO name, user's role, group and their capabilities. Then the broker is responsible for querying the VO policy engine about the user's policy and resources based on the VO information. The VO policy engine obtains the user policy and resource list from the VOMS repository according to the conditions in the queries from the resource broker. Once the user policy and resource information have been sent back to the resource broker, it can schedule the jobs based on the user policy defined by the VO and distribute the work unit to the various resources belonging to that VO.

4.5. Experiment preparation and execution

Preparation of a virtual experiment is managed via a Web page that allows the user to select and adjust various parameters including specified values and permitted ranges of values for their selected kidney models. The information is parsed and the Grid jobs are created dynamically for the broker to dispatch to the resources. Once users submit their experimental plans, the Gridbus Broker is responsible for scheduling the submission of those jobs generated by different parameters and monitoring their execution on remote Grid resources. We have implemented a generic monitoring service that extends the monitoring service in BioGrid [19] and is compatible with the Gridbus Broker 3.0 API.

4.6. Visualization

For users to gain a better understanding of modeling simulation outcomes, visualization facilities are crucial and essential for almost every life science application. In our KidneyGrid, we utilize existing visualization facilities provided in Victoria Partnership for Advanced Computing (VPAC), Melbourne, which is designed to extend mozCmgui [30] to render a 3D kidney image by reading an XML file conforming to a standard XML schema. As various kidney modeling programs are used it is not always possible to generate standard XML files. We therefore need to transform the diverse result formats to the standard that the visualization tool can read. XSLT [31] can be used as the transformation engine to accomplish the task and for each kidney model a stylesheet will be created to specify the rules for conversion to the standard format.

5. EVALUATION

We have successfully implemented a wrapper service for KSim, an implementation of a medullary kidney model implemented in the FORTRAN language [10]. Figure 4 shows the basic workflow required by users performing virtual experiments with this model. The KSim Factory Service and KSimService are two WSRF services deployed in the Globus Toolkit 4.0 (GT-4) container. To access the portal, authentication and authorization are required for obtaining the credentials for users utilizing the remote computing resources from the VO system. Single sign-on has been achieved by utilizing the MyProxy service via the Grid portlet. Once the user logs in into the portal with the username and password, the authentication module provided by the Grid portlet contacts the MyProxy server where the user's certificate has been stored during the user registration phase and manages the user's X509 proxy certificate. As an alternative, we have also developed a module that accepts the normal proxy certificate and issues a VOMS proxy certificate by contacting the VOMS server. We have considered the VOMS server as a single point of failure that may not always be available, and have allowed that the user can still choose to use the normal proxy certificate without changing any system behavior. Once users have gained the access to the portal, an experiment plan page is rendered for them, in which the user can change the parameters accepted by KSim, as demonstrated in Figure 5. The user can specify values for each parameter or make generic choices for parameters to simulate a range of values for that parameter at one time, and the KSim model itself allows users to modify up to three parameters for a new run. As soon as the user submits the experimental plan, the Gridbus Broker commences scheduling and managing the jobs generated from different parameter settings. Since an experiment is unlikely to be completed immediately, the user is able to check the status of the simulation using the execution monitor, as shown in Figure 6. This is a user-friendly facility to provide feedback of the simulation run on the remote Grid resources. It provides not only the current status of each job (simulation) on the

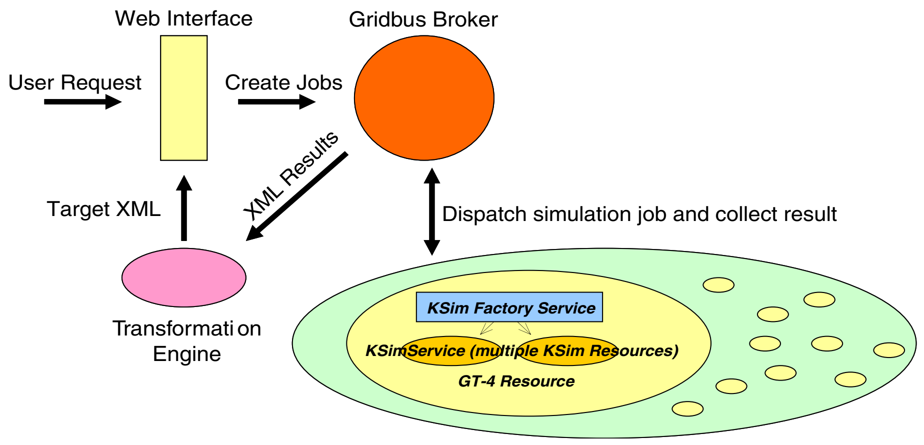
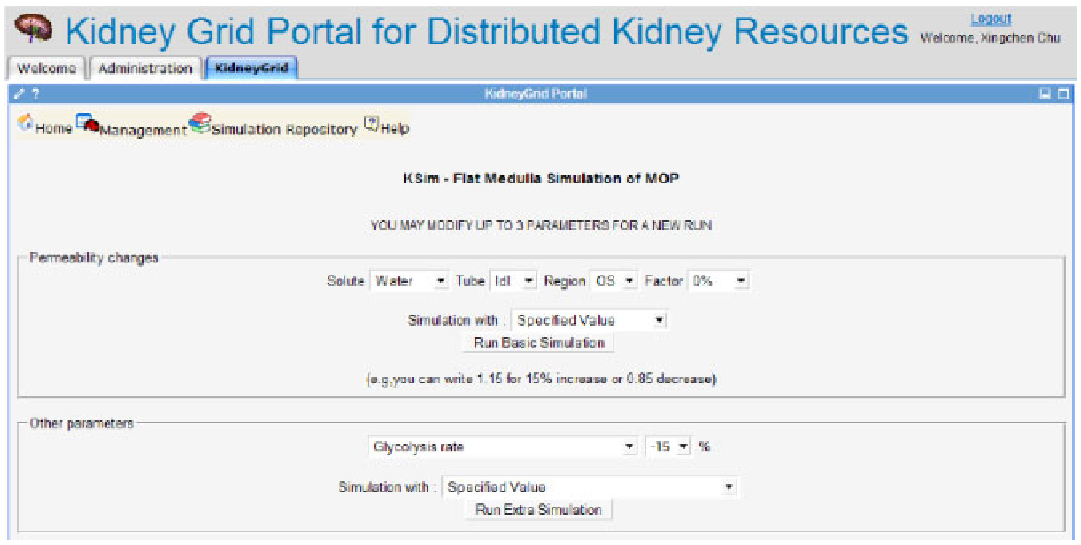


Figure 4. Workflow of KSim service.



KSim - Flat Medulla Simulation of MOP

YOU MAY MODIFY UP TO 3 PARAMETERS FOR A NEW RUN

Permeability changes

Solute: Water Tube: ldl Region: OS Factor: 0%

Simulation with: Specified Value

Run Basic Simulation

(e.g.you can write 1.15 for 15% increase or 0.85 decrease)

Other parameters

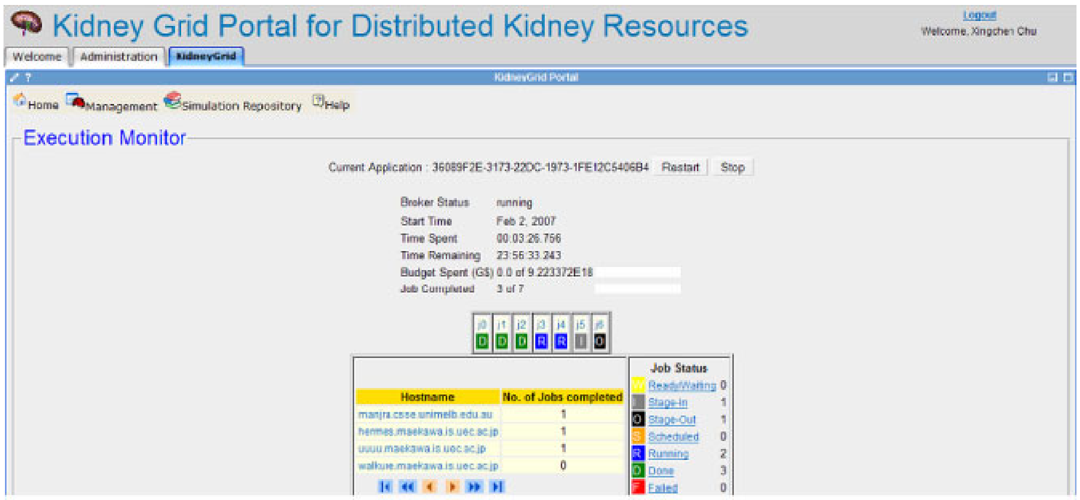
Glycolysis rate -15 %

Simulation with: Specified Value

Run Extra Simulation

Figure 5. KSim parameter settings.

remote resources but also the resource information as well as the time and budget spent for the experiment if applicable (user has specified the QoS parameters). The time remaining field shows the remaining time to the current deadline set by the user and the budget spent field illustrates the budget usage. Furthermore, any failed jobs can be reset by the user and rescheduled to available resources again. The user can then interact with the simulation results using the visualization applet shown in Figure 7, which was adopted from an applet developed for KSim [32]. The applet loads



Kidney Grid Portal for Distributed Kidney Resources

Logout
Welcome, Xingchen Chu

Welcome | Administration | KidneyGrid

KidneyGrid Portal

Home | Management | Simulation Repository | Help

Execution Monitor

Current Application : 36089F2E-3173-22DC-1973-1FE12C5406B4 Restart Stop

Broker Status	running
Start Time	Feb 2, 2007
Time Spent	00:03:26.756
Time Remaining	23:56:33.243
Budget Spent (G$)	0.0 of 9.223372E18
Job Completed	3 of 7

j0	j1	j2	j3	j4	j5	j6
D	D	D	R	R	I	O

Hostname	No. of Jobs completed
manjra.csse.unimelb.edu.au	1
hermes.maekawa.is.uec.ac.jp	1
uuuu.maekawa.is.uec.ac.jp	1
walkure.maekawa.is.uec.ac.jp	0

Job Status		
W	Ready/Waiting	0
I	Stage-In	1
O	Stage-Out	1
S	Scheduled	0
R	Running	2
D	Done	3
F	Failed	0

Figure 6. Execution monitor.

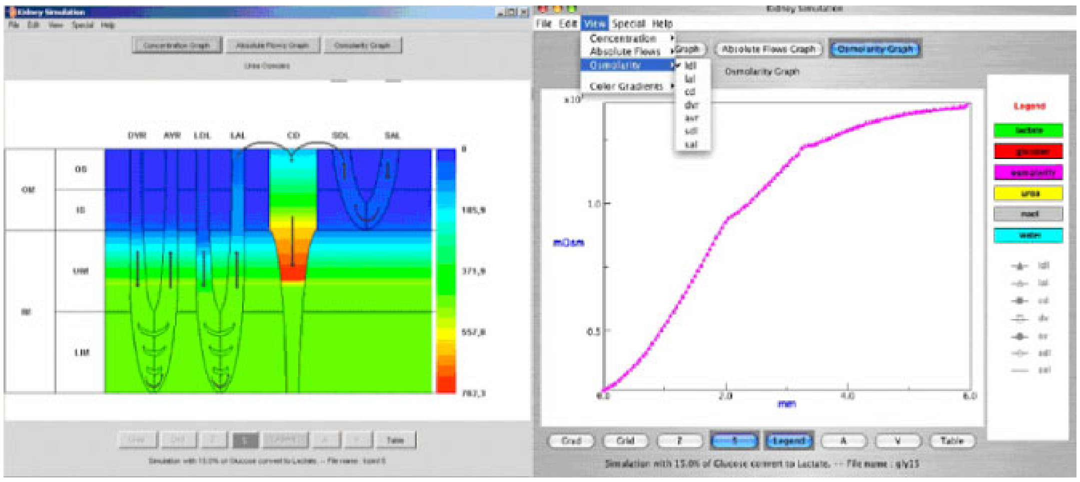


Figure 7. KSim visualization applet.

the chosen set of simulation results in the form of an XML file and permits the user to display them as x–y plots or as a color-gradient diagram of the medullary structures.

As KSim accepts four basic types of parameter including solute, tube, region and factor, and extra special cases with factors, the possible simulation cases can be expressed as the equation

$$T = C_s^1 \times C_t^1 \times C_r^1 \times C_{fn}^1 + C_x^1 \times C_{fx}^1$$

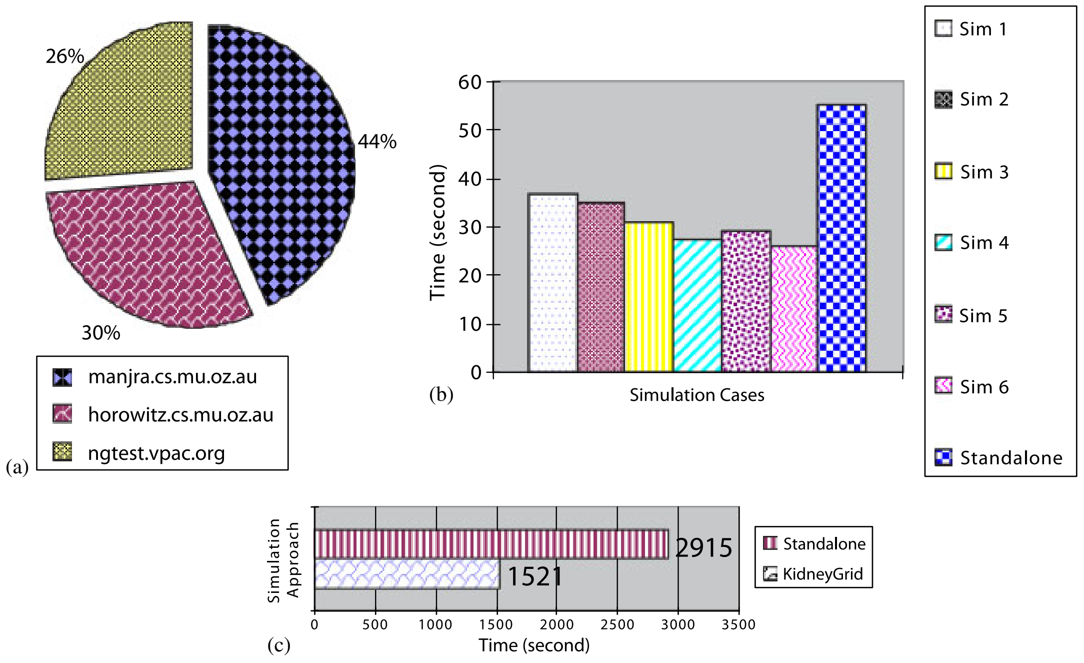


Figure 8. Evaluation of Grid jobs over resources: (a) job completion rate; (b) average job execution time; and (c) total job execution time.

where T is the total number of cases; s, t, r, x stand for solute, tube, region and extra parameter, respectively; f_n and f_x are the acceptable factors for basic and extra simulation. Depending on the value for each parameter specified in KSim, the experiment described in this example may generate a total of 513 simulation cases. To test the system when performing multiple simulations, we deployed the KSim WSRF wrapper service onto three computer servers each of which has a GT-4 container installed on each server. As our portal is only capable of generalizing one parameter per experiment, we have conducted six experiments producing values for one parameter at a time, thus generating 53 simulation cases in total. The Gridbus Broker then scheduled 53 jobs (one for each simulation) over three Grid resources.

Figure 8(a) demonstrates the jobs completed in each of the three servers. Two of the servers, ‘horowitz’ and ‘ngtest’, have a similar proportion of jobs completed (28% each) compared with the third server ‘manjra’, which has completed 44%. This is because we have configured a working queue with a limit of four jobs on ‘manjra’, which means there were four jobs running in parallel on ‘manjra’ once scheduled. According to Figure 8(b), we have conducted six parameter sweep simulations, the average execution of each simulation using our KidneyGrid infrastructure over six simulation cases was about half of the execution time expected if we had run the simulation sequentially on one machine. This result demonstrates that the parallel, distributed approach reduced by over 50% the time for computation of the selected KSim simulation. Figure 8(c) also shows that the total execution time has been decreased by about 50% using our approach. The execution time in our experiment includes all the aspects of network latency, broker overhead and actual job execution on the resource. The overall time can be further reduced by (i) running the job manager system such as PBS with queueing supports; (ii) reducing the overhead (10–12 s for each job) of the broker to schedule and (iii) adding more Grid resources that can host the simulation models. Our experiment shows that our KidneyGrid provides improved throughput from experiment plan to result visualization by distributing kidney modeling simulations over Grid resources that have substantially reduced the complexity of the usage and the computation time for batch experiments.

6. CONCLUSION AND FUTURE WORK

In this paper, a life science infrastructure to Grid-enable distributed legacy applications and resources utilizing SOA is presented as part of a larger international project, the 'Renal Physiome'. We focus on Grid-enabling various kidney models within this infrastructure, demonstrating the ability to compose, schedule, monitor and visualize the result of the application. The use of the Grid resource broker not only simplifies the development of application, credential and resource management but also decouples our platform from the underlying Grid middleware. WSRF makes the state-oriented Web services applicable for Grid application, which provides a lightweight solution to integrate legacy applications by implementing service wrappers for them. We have successfully implemented

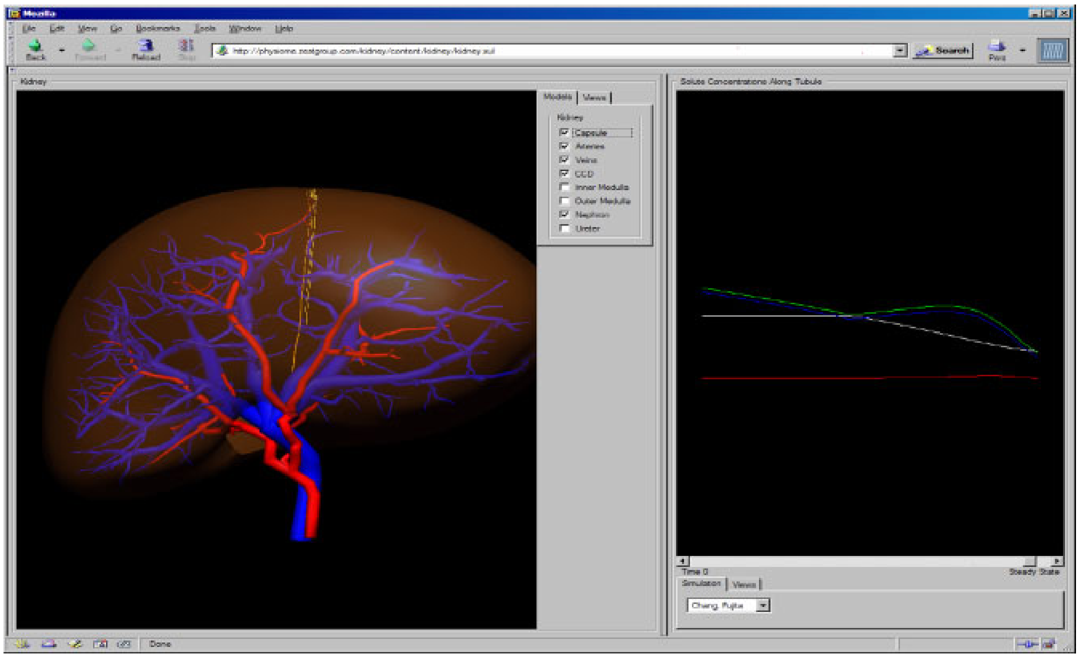


Figure 9. 3D kidney visualization tool.

the KSim model on top of our infrastructure and deployed the KSim services on several remote resources including servers located within Australia and also four overseas workstations located in Japan. We have also extended the original Gridbus broker to support VO-based authentication and scheduling on top of the VOMS. In addition, the 3D visualization facility mozCmGUI [30], as shown in Figure 9, was investigated and extended by VPAC to support the KSim model.

It should be noted that the project is still in the early stages of development, and much still needs to be accomplished. Wrappers are required for the kidney models on our list so that the successful development and deployment of the wrapper service for the KSim model can be applied to other kidney models. We also need to create stylesheets to format each model result into the standard representation and the existing 3D visualization tool, after user evaluation is integrated into the platform. As XSLT only works with XML input, we will utilize other transformation tools to handle non-XML input generated by the legacy kidney models. Other ongoing work includes the investigation of the complex scheduling algorithms for the broker based on the VOMS service and user policy we have implemented within our platform.

ACKNOWLEDGEMENTS

The authors would like to express their gratitude to members of the Gridbus Project for sharing their thoughts on Grid-enabling the kidney models. We thank Kyong Hoon Kim for his contribution toward implementing VO-based authorization and scheduling and configuring VOMS server and Hussein Gibbins for his comments and sharing his experiences about how to use the broker services to develop a portal for distributed molecular docking. We thank the ARC (Australia Research Council) for their partial support to this project through their SRI (Special Research Initiative) on e-Research. We also thank DEST (Department of Education, Science and Training) for their support through International Science Linkage Program for extending Gridbus technologies.

REFERENCES

1. Chang H, Fujita T. A numerical model of the renal distal tubule. *American Journal of Physiology—Renal Physiology* 1999; **276**(6 Pt 2):F931–F951.
2. Weinstein AM. A mathematical model of the rat proximal tubule. *American Journal of Physiology—Renal Physiology* 1986; **250**(5 Pt 2):F860–F873.
3. Thomas SR, Dagher G. A kinetic model of rat proximal tubule transport—load dependent bicarbonate reabsorption along the tubule. *Bulletin of Mathematical Biology* 1994; **56**(3):431–458.
4. Layton HE, Pitman EB, Moore LC. Bifurcation analysis of TGF-mediated oscillations in SNGFR. *American Journal of Physiology* 1991; **261**(5 Pt 2):F904–F919.
5. Holstein-Rathlou NH, Wagner AJ, Marsh DJ. Tubuloglomerular feedback dynamics and renal blood flow autoregulation in rats. *American Journal of Physiology—Renal Physiology* 1991; **260**:F53–F68.
6. Deen WM, Robertson CR, Brenner BM. A model of glomerular ultrafiltration in the rat. *American Journal of Physiology—Renal Physiology* 1972; **223**(5):1178–1183.
7. Foster DM, Jacquez JA. Comparison using central core model of renal medulla of the rabbit and rat. *American Journal of Physiology—Renal Physiology* 1978; **234**(5):F402–F414.
8. Stephenson JL, Mejia R, Tewarson RP. Model of solute and water movement in the kidney. *Proceedings of the National Academy of Sciences of the United States of America* 1976; **73**(1):252–256.
9. Lory P. Effectiveness of a salt transport cascade in the renal medulla: Computer simulations. *American Journal of Physiology—Renal Physiology* 1987; **252**(6):F1095–F1102.
10. Hervy S, Thomas SR. Inner medullary lactate production and urine-concentrating mechanism: A flat medullary model. *American Journal of Physiology—Renal Physiology* 2003; **284**(1):F65–F81.
11. Foster I, Kesselman C. *The Grid*: *Blueprint for a Future Computing Infrastructure*. Morgan Kaufmann Publishers: Los Altos, CA, 1999.

12. Foster I, Kesselman C. The Globus project: A status report. *Proceedings of IPPS/SPDP'98 Heterogeneous Computing Workshop*. IEEE Computer Society Press: U.S.A., 1998.
13. Venugopal S, Buyya R, Winton L. A Grid service broker for scheduling e-Science applications on global data Grids. *Concurrency and Computation*: *Practice and Experience* 2006; **18**(6):685–699.
14. Alfieri R, Cecchini R, Ciaschini V, dell'Agnello L, Frohner A, Gianoli A, Lorentey K, Spataro F. Voms: An authorization system for virtual organizations. *Proceedings of the 1st European Across Grids Conference*, Santiago de Compostela, February 2003.
15. Baru C, Moore R, Rajasekar A, Wan M. The SDSC storage resource broker. *Proceedings of CASCON'98*, Toronto, Canada, November 1998.
16. QKDB. http://physiome.ibisc.fr/qkdb/ [October 2006].
17. Bassingthwaighte JB. Strategies for the Physiome project. *Annual Biomedical Engineering* 2000; **28**:1043–1058.
18. Beeson B, Melnikoff S, Venugopal S, Barnes DG. A portal for Grid-enabled physics. *Proceedings of the 2005 Australasian Workshop on Grid Computing and e-Research*, Newcastle, Australia, January 2005.
19. Gibbins H, Nadiminti K, Beeson B, Chhabra R, Smith B, Buyya R. The Australian BioGrid portal: Empowering the molecular docking research community. *Proceedings of APAC 2005 Conference*, Gold Coast, Australia, September 2005.
20. Buyya R, Branson K, Giddy J, Abramson D. The virtual laboratory: Enabling molecular modeling for drug design on the world wide Grid. *Concurrency and Computation*: *Practice and Experience* 2003; **15**(1):1–25.
21. Kolbe S, Ma T, Liu W, Soh WS, Buyya R, Egan G. A platform for distributed analysis of neuroimaging data on global Grids. *Proceedings of the 1st International Conference on e-Science and Grid Computing*, Melbourne, Australia, 5–8 December 2005.
22. Zhou Z, Wang F, Todd BD. Development of chemistry portal for Grid-enabled molecular science. *Proceedings of the 1st International Conference on e-Science and Grid Computing*, Melbourne, Australia, 5–8 December 2005.
23. Sudholt W, Baldridge K, Abramson D, Enticott C, Garic S. Application of Grid computing to parameter sweeps and optimizations in molecular modeling. *Proceedings of the 4th International Conference on Computational Sciences*, Krakow, Poland, 6–9 June 2004.
24. Lloyd CM, Halstead MDB, Nielsen PF. CellML: its future, present and past. *Progress in Biophysics and Molecular Biology* 2004; **85**(2–3):433–450.
25. Lonie AJ, Stevens C, Harris P. LB520: Computer modelling of kidney function. *Experimental Biology 2004*, Washington, DC, 2004.
26. Novotny J, Russell M, Wehrens O. GridSphere: A portal framework for building collaborations. *Concurrency and Computation*: *Practice and Experience* 2004; **16**(5):503–513.
27. Basney J, Humphrey M, Welch V. The MyProxy Online Credential Repository. *Software: Practice and Experience* 2005; **35**(9):801–816.
28. Foster I, Czajkowski K, Ferguson D, Frey J, Graham S, Maguire T, Snelling D, Tuecke S. Modeling and managing state in distributed systems: The role of OGSI and WSRF. *Proceedings of the IEEE* 2005; **93**:604–612.
29. Luther A, Buyya R, Ranjan R, Venugopal S. Alchemi: A .NET-based enterprise Grid computing system. *Proceedings of the 6th International Conference on Internet Computing*, Las Vegas, U.S.A., 27–30 June 2005.
30. mozCmgui. http://www.cmiss.org/ReleaseCenter/mozcmgui [May 2006].
31. XSLT. http://www.w3.org/TR/xslt [May 2006].
32. KSim. http://www.lami.univ-evry.fr/~srthomas/kidneysim/ [May 2006].