Joint International Conference on Supercomputing in Nuclear Applications and Monte Carlo 2010 (SNA + MC2010)
Hitotsubashi Memorial Hall, Tokyo, Japan, October 17-21, 2010

A Portable Grid-Enabled Computing System for A Nuclear Material Study

Yuichi TSUJITA [1*], Tatsumi ARIMA [2],
Takayuki TATEKAWA [3] and Yoshio SUZUKI [3]

[1] *Faculty of Engineering, Kinki University, 1 Umenobe, Takaya, Higashi-Hiroshima, Hiroshima 739-2116, Japan*
[2] *Faculty of Engineering, Kyushu University, 744 Motooka, Nishi-ku, Fukuoka 819-0395, Japan*
[3] *Center for Computational Science and e-Systems, Japan Atomic Energy Agency, 6-9-3 Higashi-Ueno, Taito-ku, Tokyo 110-0015, Japan*

We have built a portable grid-enabled computing system specialized for our molecular dynamics (MD) simulation program to study Pu material easily. Experimental approach to reveal properties of Pu materials is often accompanied by some difficulties such as radiotoxicity of actinides. Since a computational approach reveals new aspects to researchers without such radioactive facilities, we address an MD computation. In order to have more realistic results about e.g., melting point or thermal conductivity, we need a large scale of parallel computations. Most of application users who don't have supercomputers in their institutes should use a remote supercomputer. For such users, we have developed the portable and secured grid-enabled computing system to utilize a grid computing infrastructure provided by ITBL (Information Technology Based Laboratory). This system enables us to access remote supercomputers in the ITBL system seamlessly from a client PC through its graphical user interface (GUI). Typically it enables seamless file accesses on the GUI. Furthermore monitoring of standard output or standard error is available to see progress of an executed program. Since the system provides fruitful functionalities which are useful for parallel computing on a remote supercomputer, researchers can concentrate on their researches.

KEYWORDS: Pu recycle, molecular dynamics simulation, grid computing, Java Native Interface, client computing system

I. Introduction

Recently, Pu recycle is focused for effective use of nuclear fuel (e.g., IFNEC mission [1)]). In the advanced fuel cycle, MOX and inert matrix fuels such as ZrO_2-PuO_2 are expected for effective burning Pu. However, research works about it are restricted in limited research institutes because of expensive experimental facilities which can manage radiotoxicity of Pu. Therefore we have addressed computer simulations not only to minimize costs for experimental researches but also to have deep insights through analysis of it by using an MD simulation. Due to the lack of a supercomputer in user's research groups, we need to use a remote supercomputer with secured connections. Therefore we have been utilizing a grid computing environment provided by the ITBL project [2)] to ensure a secured connection to a remote supercomputer seamlessly.

The ITBL project is one of the national grid projects in Japan. Its computing infrastructure (an ITBL system) provides a seamless computing environment and huge amount of computational resources by connecting many parallel computers of different institutes via SINET3 [3)] with a secured communication infrastructure. The ITBL system provides fruitful general purpose server-side software tools on a portal site in which a user is registered. We can utilize the tools such as a file manager, a program execution manager, and a terminal manager without any attention to heterogeneity in underlying communication systems and locations where computational resources are. Besides a client API library is provided to build an application-oriented grid application which runs on a client PC.

Application users are non-expert in grid computing and their target is computer simulations. Therefore we built an application-oriented GUI computing system for our simulation program by using an ITBL Java API as reported in [4)] to orchestrate computation on a remote supercomputer in the ITBL system and a native visualization program on a client PC seamlessly although the ITBL system provides a visualization system which cooperates with AVS/Express [5)]. Its GUI removed complexity of combination of many parameter survey runs and users can operate computations without mistakes. However, it lacked usability in remote file accesses and monitoring standard output and standard error. A pop-up window was provided to select a directory or a file in the previous version. Users felt a gap in manipulations between a main window and the pop-up window. Copying and deleting directories and files were not available. Besides the lack of monitoring prevented users to check whether a program was running correctly or not.

Regarding to the lacks and termination of the Java API delivery support, we have newly designed a computing system based on the previous one by using the C++ library and a Java Native Interface (JNI) on a Windows PC. The new system can indicate standard output and standard error on demand during execution of a program. Users can check whether the program is running correctly or not easily and

*Corresponding Author, E-mail:tsujita@hiro.kindai.ac.jp

seamlessly without waiting for completion of a program. Furthermore, it can assist seamless remote file manipulations such as copying and deleting on the same GUI window dedicated for job execution. Since contents of a directory tree and a file list are updated after each operation, users can easily know progress of each operation.

In the rest of this paper, sections II and III briefly describe an MD computation and the ITBL system, respectively. Section IV explains motivation and details of the computing system. Related work is discussed in section V, followed by conclusion in section VI.

II. Molecular Dynamics Program

MD computations have been performed by a parallel version of MXDORTO named MXDORTOP [6] written in Fortran77 to have simulation results about e.g., thermal conductivity or melting point. Its computation task by the Ewald method is parallelized by using Message Passing Interface (MPI) [7] because the computation is the most compute intensive part. **Figure 1** shows a flowchart of the program code.

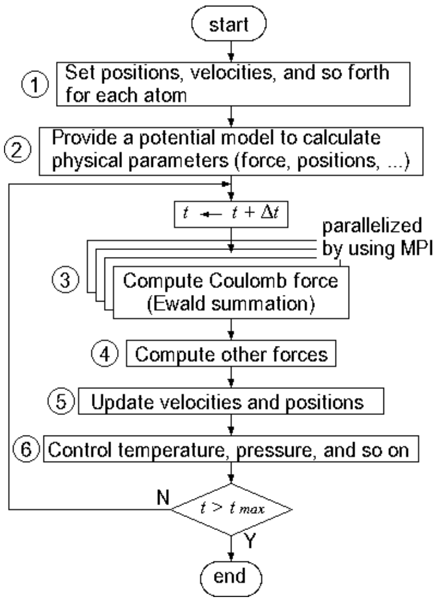

start
① Set positions, velocities, and so forth for each atom
② Provide a potential model to calculate physical parameters (force, positions, ...)
$t \leftarrow t + \Delta t$
parallelized by using MPI
③ Compute Coulomb force (Ewald summation)
④ Compute other forces
⑤ Update velocities and positions
⑥ Control temperature, pressure, and so on
N
$t > t_{max}$
Y
end

Fig. 1 Flowchart of a parallelized MD code, MXDORTOP

This program reads some parameters such as initial positions and velocities of every atom, the number of atoms (1), and a potential model from several parameter files at start-up (2). In each step, parallelized subroutines by using the Ewald method calculate forces for every atom (3), followed by calculations of other forces by a root process (rank = 0 in an MPI program) (4). Based on the forces, velocities and positions are updated (5). At the final stage, temperature, pressure, and so on are updated according to the calculated results (6). The program outputs intermediate calculation results about positions of all atoms, temperature, and so forth at each given time step. Some of the outputs are used for visualization of atoms or post-calculation of heat capacity and thermal conductivity.

Since our target material is assumed to be a partially ionic crystal, we selected a Busing-Ida-Gilbert type partially ionic pair potential. At this moment, we adopt Pu-O and U-O ionic pairs for our simulation research works. We assume the partial ionic charges as 67.5 % of their formal charges. Potential parameters were obtained from the works described in [8-9].

III. ITBL System

ITBL project is a national project as one of the e-Japan Priority Policy Program. This project was launched in April 2001 by six institutes: the National Institute for Materials Science, the National Research Institute for Earth Science and Disaster Prevention, Japan Aerospace Exploration Agency, the Institute of Physical and Chemical Research, Japan Science and Technology Agency, and Japan Atomic Energy Agency and has been carried out until March 2006 as five years' plan. Project target is establishing virtual laboratories where many researchers in various disciplines can collaborate in developing a large scale of computer simulation systems by utilizing computer resources connected by high-speed network. At the end of the project, 681 researchers from 90 organizations participated. At this moment, this project is maintained by the five institutes except the Japan Science and Technology Agency.

An ITBL system provides a flat computing environment by using its secured communication software such as an RPC system named starpc [10]. Every computing resource is managed in the logical unit named “site” which is protected by a firewall from Internet. Every site comprises ITBL servers for secured access from/to Internet. The ITBL servers consist of three servers: front, data, and proxy servers. The front and data servers are located in DMZ (DeMilitarized Zone) in order to protect at least computing resources such as supercomputers inside a site. The front server manages connections between user's client PC and ITBL systems. It also manages seamless connections between sites. The data server handles information to manage ITBL systems including information of ITBL users. The proxy server is for connections between the front server and supercomputers inside a site. Every computing resource is accessible via the ITBL servers after authentication.

A certificate authority generates an X.509 certificate for every ITBL user’s single sign-on in authentication and authorization. The certificate is deployed in both a client PC and ITBL servers of a portal site in which the user is assigned. Later the user can access a portal site through a web browser by using an https protocol. The user can access remote supercomputers in which the user has an account not only inside the portal site but also in other sites. Authorization for the other sites is realized by the front server of the portal site. A portal site menu window provides many kinds of server-side software tools such as a file manager and a program execution manager. Moreover, a client API library has been developed to build an application-oriented grid system which runs on a client PC. The library has been built on top of the ITBL's secured and seamless communication software stack to realize the same authentication and authorization. At this moment the library supports C++ for Windows with the help of Microsoft

Visual C++ and C for Linux.

IV. Client GUI computing system for Nuclear Material Engineering

We have proposed a large scale of computation by using a remote supercomputer in the ITBL system. However, we have faced usability problems and difficulty in setting many parameters for MXDORTOP as described in Introduction. For parameter settings, they need to arrange an input file for the program in each run. Molecule structural data files should be moved manually from a supercomputer to a client PC on which visualization is carried out. Furthermore, application users want to check progress of a running program without deep understanding in a grid computing.

We have redesigned our client GUI computing system based on the previous system by using the C++ API library and JNI. We have selected Java to build an upper layer software stack which faces to application users because Java provides fruitful GUI and it is platform independent. This system provides the following functions:

(1) Job management

This function assists to select a simulation program and parameter settings for the program. It also supports seamless job submission and job status monitoring.

(2) Seamless data file manipulations

A data file is seamlessly uploaded and downloaded by using this function. Copying and deleting data files are also available. Besides, monitoring contents of a remote file is realized.

(3) Visualization support for remote calculated data by using a native visualization software

Seamless parallel computations followed by visualization of calculated results by a native visualization program are supported.

Figure 2 depicts architecture of the client GUI system connecting to ITBL servers.

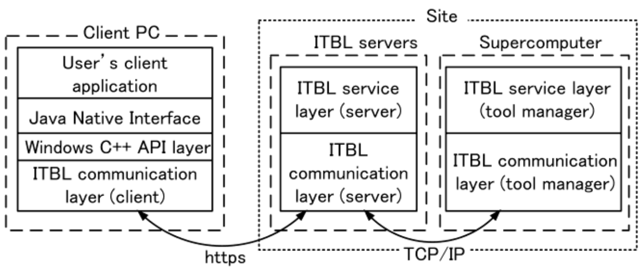

Client PC
User's client application
Java Native Interface
Windows C++ API layer
ITBL communication layer (client)
Site
ITBL servers
ITBL service layer (server)
ITBL communication layer (server)
Supercomputer
ITBL service layer (tool manager)
ITBL communication layer (tool manager)
https
TCP/IP

Fig. 2 Architecture of a client GUI system connecting to ITBL servers

The Windows C++ API layer of the client library is on top of the ITBL's secured communication layer. The library has many interfaces to utilize ITBL's functionalities such as a file manager, a program execution manager, a resource information service manager. On top of the library, we have deployed a JNI layer to use the C++ library from a Java application. Requests from a client PC to supercomputers are transferred by using an ITBL's RPC named starpc. A communication path between the client PC and the ITBL servers is established by using an https protocol for secured connections. On the other hand, TCP/IP is used for communications between ITBL servers and supercomputers because they are inside the same site.

Figure 3 illustrates a task flow of the client GUI system.

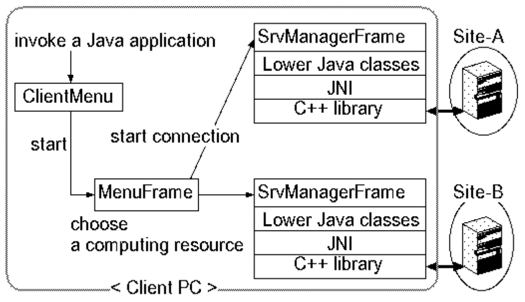


Fig. 3 Task flow of a developed client GUI system

Firstly, a Java application built by a *ClientMenu* class is invoked. Then the class invokes a menu window built by a *MenuFrame* class. In the menu window, a user can start a new menu window built by an *SrvManagerFrame* class for a target supercomputer. The window starts a connection to a target supercomputer by using the ITBL's secured communication software. We can access every supercomputer in the registered site seamlessly and start job execution once the window succeeds the connection. Details of this system are explained in the following subsections.

IV-1. A common GUI menu window

A thing to do at first is execution of the GUI system on a client PC by using a Java start-up command with the help of an own shared library for JNI. **Figure 4** shows a start-up menu window, where a user is required to give a pass-phrase for an X.509 certificate to connect a portal site.

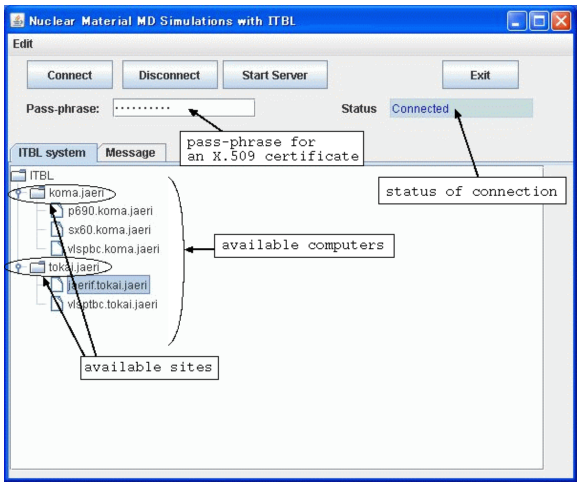
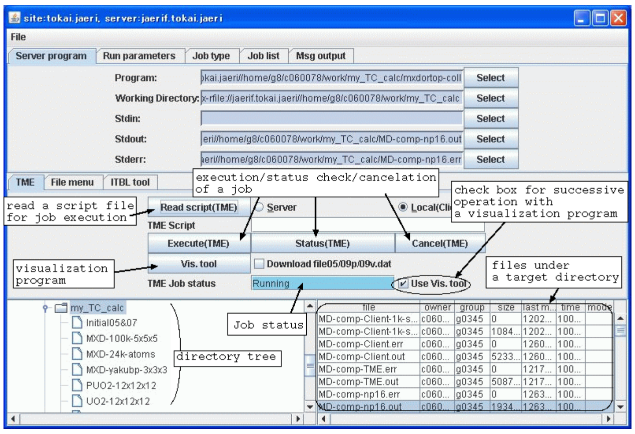


Fig. 4 Start-up menu window of a client GUI system

Later, we can see available sites and associated computers. We need to describe the URL of a registered site (portal site) in a configuration file. The location of the X.509 certificate can be specified in the same file. The client GUI system reads the file at its start-up to activate such parameters. Only the pass-phrase must be given on the start-up window for security reason. A user can modify the parameters in the file by selecting a menu, “Preferences,” in a pull-down menu, “File.” Later the GUI system can connect to the portal site by pressing a button, “Connect.” Once authentication succeeds, the start-up window indicates available computing resources in a tree structure manner. Here two different sites (“koma.jaeri” and “tokai.jaeri”) are indicated as folder icons under the root icon (“ITBL”) in a tree structure manner. Available computers including supercomputers in each site are indicated under a corresponding site icon. Once a user selects one of the computers, pressing a button, “Start Server,” invokes a new GUI window for job execution.

IV-2. Job execution menu

A job execution menu window built by an *SrvManagerFrame* class is initiated for a target supercomputer as shown in **Figure 5**.



site:tokai.jaeri, server:jaerif.tokai.jaeri
File
Server program
Run parameters
Job type
Job list
Msg output
Select
TME
File menu
ITBL tool
execution/status check/cancelation of a job
read a script file for job execution
check box for successive operation with a visualization program
Read script(TME)
Server
Local(Cli
TME Script
Execute(TME)
Status(TME)
Cancel(TME)
visualization program
Vis. tool
files under a target directory
Job status
directory tree

Program: kai.jaeri//home/g8/c060078/work/my_TC_calc/mxdortop-coll
Working Directory: x-rfile://jaerif.tokai.jaeri//home/g8/c060078/work/my_TC_calc
Stdin:
Stdout: eri//home/g8/c060078/work/my_TC_calc/MD-comp-np16.out
Stderr: aeri//home/g8/c060078/work/my_TC_calc/MD-comp-np16.err

Download file05/09p/09v.dat
TME Job status: Running
[✓ Use Vis. tool]

my_TC_calc
Initial05&07
MXD-100k-5x5x5
MXD-24k-atoms
MXD-yakubp-3x3x3
PUO2-12x12x12
UO2-12x12x12

file	owner	group	size	last m...	time	mode
MD-comp-Client-1k-s...	c060...	g0345	0	1202...	100...	
MD-comp-Client-1k-s...	c060...	g0345	1084...	1202...	100...	
MD-comp-Client.err	c060...	g0345	0	1260...	100...	
MD-comp-Client.out	c060...	g0345	5233...	1260...	100...	
MD-comp-TME.err	c060...	g0345	0	1217...	100...	
MD-comp-TME.out	c060...	g0345	5087...	1217...	100...	
MD-comp-np16.err	c060...	g0345	0	1263...	100...	
MD-comp-np16.out	c060...	g0345	1934...	1263...	100...	

Fig. 5 GUI window for a target computer

This GUI window is initiated as a new thread, thereby it can run in parallel with the start-up window. In the middle of this window, there are a “TME” tab menu which has buttons for job controls and visualization of calculated results as follows:

(1) Read script(TME)
(2) Execute(TME)
(3) Status(TME)
(4) Cancel(TME)
(5) Vis. Tool

The program execution manager in an ITBL portal menu supports exporting a script file based on a task flow menu named TME. In order to reuse the file, this system imports it from a remote supercomputer or a client PC by activating check boxes, “Server” or “Local (Client PC),” respectively. Those operations are realized with the help of the ITBL client API of a program execution manager. The first button, “Read script(TME),” starts reading a job execution script file for an ITBL system from the selected server machine or user's client PC. Its file name is indicated under the button. Scripts in the file are used for job execution. The second one, “Execute(TME),” execute a simulation program by using the script file and the job is submitted to a batch system or interactively executed. Pressing the third one, “Status(TME),” displays job status on a text-field, “TME Job status,” in the middle of this menu. Submitted or running job can be canceled by the fourth button, “Cancel(TME).” For visualization of simulation results, VESTA [11] is used at this moment. The fifth button, “Vis. tool,” starts the visualization software. Moreover, it is automatically started after completion of a simulation program by activating a check-box, “Use Vis. tool.”

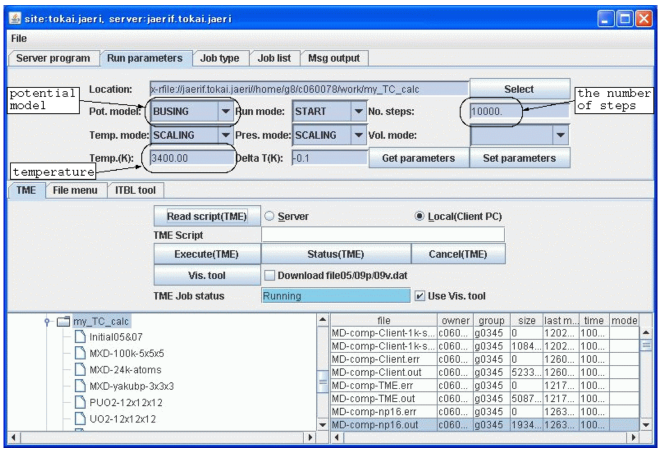
Directory tree and file list are indicated on the lower part of the window from this version. We can select a target file or directory from here in e.g., specification of a simulation program or a working directory, copying and moving files, and so on by using function buttons in a “file menu” tab next to the “TME” tab. A user can also download or upload between a client PC and a supercomputer via secured connections by selecting “Download” or “Upload” in a “File” pull-down menu on the upper of it, respectively.

The window has tab-menus which handle the following items in the upper location of the window:

(1) Server program
(2) Run parameters
(3) Job type
(4) Job list
(5) Msg output (Message output)

The first menu, “Server program,” in Figure 5 is designed to control execution of a simulation program. A user can specify a full-path name of a simulation program, a working directory, a standard input, a standard output, and a standard error. Once a user selects a target program or a file in the directory tree or the file list, pressing a “Select” button of each parameter validates the target directory or the file.

Figure 6 shows a tab menu, “Run parameters.”



Location: x-rfile://jaerif.tokai.jaeri//home/g8/c060078/work/my_TC_calc
Pot. model: BUSING
Run mode: START
No. steps: 10000.
Temp. mode: SCALING
Pres. mode: SCALING
Vol. mode:
Temp.(K): 3400.00
Delta T(K): -0.1

Download file05/09p/09v.dat
TME Job status: Running
[✓ Use Vis. tool]

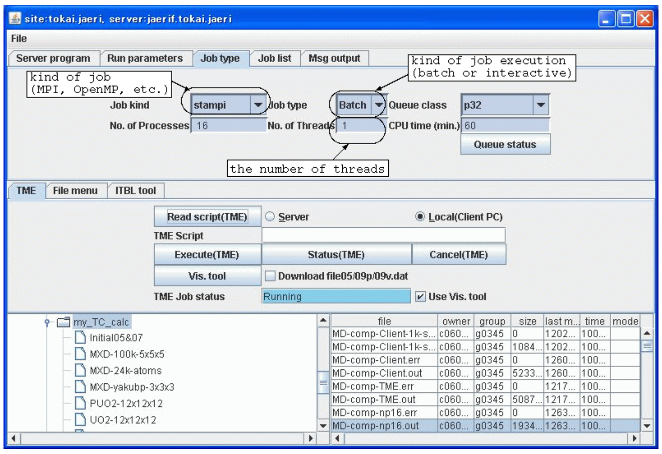
my_TC_calc
Initial05&07
MXD-100k-5x5x5
MXD-24k-atoms
MXD-yakubp-3x3x3
PUO2-12x12x12
UO2-12x12x12

file	owner	group	size	last m...	time	mode
MD-comp-Client-1k-s...	c060...	g0345	0	1202...	100...	
MD-comp-Client-1k-s...	c060...	g0345	1084...	1202...	100...	
MD-comp-Client.err	c060...	g0345	0	1260...	100...	
MD-comp-Client.out	c060...	g0345	5233...	1260...	100...	
MD-comp-TME.err	c060...	g0345	0	1217...	100...	
MD-comp-TME.out	c060...	g0345	5087...	1217...	100...	
MD-comp-np16.err	c060...	g0345	0	1263...	100...	
MD-comp-np16.out	c060...	g0345	1934...	1263...	100...	

Fig. 6 GUI window for parameters of a simulation program

Here we can specify parameters for MD computations such as a potential model, the number of calculation steps, temperature, pressure, and so on. Modified parameters are applied to a parameter file of the MD code on the target supercomputer by pressing a button, “Set parameters.” In addition, a user can get all parameters, which are written already in a parameter file, by pressing a button, “Get parameters.”

The third tab menu, “Job type,” is for parameters of job executions as shown in **Figure 7**.



Job kind: stampi
Job type: Batch
Queue class: p32
No. of Processes: 16
No. of Threads: 1
CPU time (min.): 60

Server [✓ Local(Client PC)]
TME Script
Download file05/09p/09v.dat
TME Job status: Running
[✓] Use Vis. tool

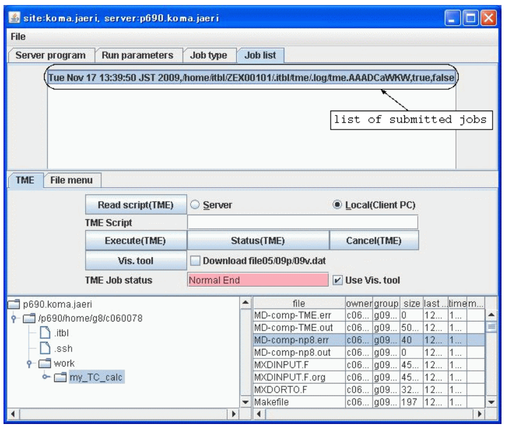
my_TC_calc
- Initial05&07
- MXD-100k-5x5x5
- MXD-24k-atoms
- MXD-yakubp-3x3x3
- PUO2-12x12x12
- UO2-12x12x12

file	owner	group	size	last m...	time	mode
MD-comp-Client-1k-s...	c060...	g0345	0	1202...	100...	
MD-comp-Client-1k-s...	c060...	g0345	1084...	1202...	100...	
MD-comp-Client.err	c060...	g0345	0	1260...	100...	
MD-comp-Client.out	c060...	g0345	5233...	1260...	100...	
MD-comp-TME.err	c060...	g0345	0	1217...	100...	
MD-comp-TME.out	c060...	g0345	5087...	1217...	100...	
MD-comp-np16.err	c060...	g0345	0	1263...	100...	
MD-comp-np16.out	c060...	g0345	1934...	1263...	100...	

Fig. 7 Job type selection window

We have serial and parallel modes for computation. Besides, OpenMP [12] is also supported for shared memory programming in addition to distributed memory parallel computing using MPI. A user can select one of them according to a simulation program. It is noticed that a simulation code is generally executed by a batch system on supercomputers while an interactive mode is selected in a test phase. A user can specify parameters associated with job execution such as an execution mode (interactive/batch), a queue class (mandatory for the batch system), the number of processes, and the number of threads in this menu.

The fourth tab menu, “Job list,” is for monitoring all submitted jobs. **Figure 8** shows an example of this menu.



```
Tue Nov 17 13:39:50 JST 2009,/home/itbl/ZEX00101/.itbl/tme/.log/tme.AAADCaWKW,true,false
```

Server [✓ Local(Client PC)]
TME Script
Download file05/09p/09v.dat
TME Job status: Normal End
[✓] Use Vis. tool

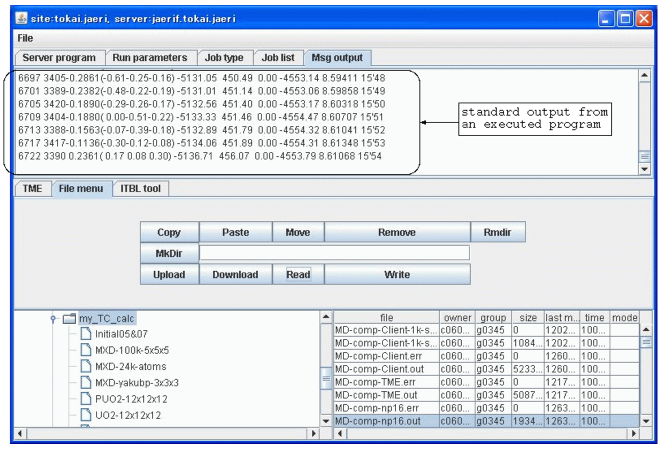
p690.koma.jaeri
- /p690/home/g8/c060078
  - .itbl
  - .ssh
  - work
    - my_TC_calc

file	owner	group	size	last ...	time	m...
MD-comp-TME.err	c06...	g09...	0	12...	1...	
MD-comp-TME.out	c06...	g09...	50...	12...	1...	
MD-comp-np8.err	c06...	g09...	40	12...	1...	
MD-comp-np8.out	c06...	g09...	0	12...	1...	
MXDINPUT.F	c06...	g09...	45...	12...	1...	
MXDINPUT.F.org	c06...	g09...	45...	12...	1...	
MXDORTO.F	c06...	g09...	32...	12...	1...	
Makefile	c06...	g09...	197	12...	1...	

Fig. 8 Job list GUI window

A status of a selected job is indicated in the middle of this program execution menu window.

The last tab menu, “Msg output,” supports monitoring standard output/error produced by a running program on demand. An example of it is shown in **Figure 9**.



```
6697 3405-0.2861(-0.61-0.25-0.16) -5131.05  450.49  0.00 -4553.14 8.59411 15'48
6701 3389-0.2382(-0.48-0.22-0.19) -5131.01  451.14  0.00 -4553.06 8.59858 15'49
6705 3420-0.1890(-0.29-0.26-0.17) -5132.56  451.40  0.00 -4553.17 8.60318 15'50
6709 3404-0.1880( 0.00-0.51-0.22) -5133.33  451.46  0.00 -4554.47 8.60707 15'51
6713 3388-0.1563(-0.07-0.39-0.18) -5132.89  451.79  0.00 -4554.32 8.61041 15'52
6717 3417-0.1136(-0.30-0.12-0.08) -5134.06  451.89  0.00 -4554.31 8.61348 15'53
6722 3390 0.2361( 0.17 0.08 0.30) -5136.71  456.07  0.00 -4553.79 8.61068 15'54
```

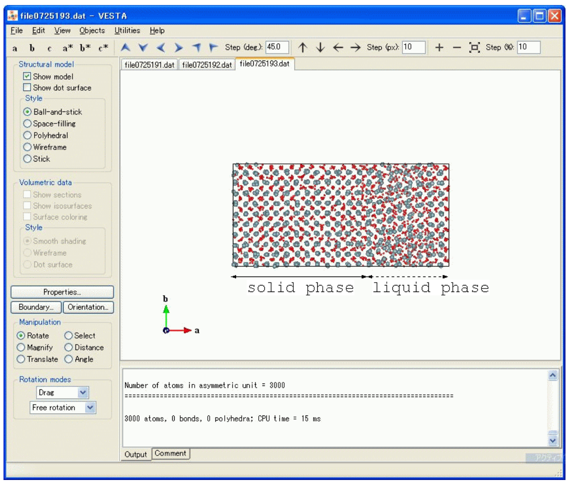
my_TC_calc
- Initial05&07
- MXD-100k-5x5x5
- MXD-24k-atoms
- MXD-yakubp-3x3x3
- PUO2-12x12x12
- UO2-12x12x12

file	owner	group	size	last m...	time	mode
MD-comp-Client-1k-s...	c060...	g0345	0	1202...	100...	
MD-comp-Client-1k-s...	c060...	g0345	1084...	1202...	100...	
MD-comp-Client.err	c060...	g0345	0	1260...	100...	
MD-comp-Client.out	c060...	g0345	5233...	1260...	100...	
MD-comp-TME.err	c060...	g0345	0	1217...	100...	
MD-comp-TME.out	c060...	g0345	5087...	1217...	100...	
MD-comp-np16.err	c060...	g0345	0	1263...	100...	
MD-comp-np16.out	c060...	g0345	1934...	1263...	100...	

Fig. 9 GUI window for a standard output

We can easily check progress of an executed program by choosing a target standard output file or standard error file and pressing a button, “Read,” on demand in the “File menu” tab on the menu window. Since every component is on the same window, users don’t feel any gaps in manipulations. If there is some strange behavior in the output, we can notify some problems in computations soon.

Once a computation finishes, calculated data on a remote supercomputer can be visualized seamlessly. **Figure 10** shows a visualized image obtained from two phase simulations with 3,000 PuO_2 atoms with 30,000 time steps.



Step (deg.): 45.0 Step (px): 10 Step (%): 10

Structural model
- [✓] Show model
- Show dot surface
- Style: [✓ Ball-and-stick] Space-filling Polyhedral Wireframe Stick

Volumetric data
- Show sections
- Show isosurfaces
- Surface coloring
- Style: Smooth shading Wireframe Dot surface

Manipulation: [✓ Rotate] Select Magnify Distance Translate Angle

Rotation modes: Drag; Free rotation

```
Number of atoms in asymmetric unit = 3000
================================================================================

3000 atoms, 0 bonds, 0 polyhedra; CPU time = 15 ms
```

Fig. 10 Visualized image by VESTA

The image was generated by the VESTA on a client PC after automatic transfer of a structural data file from a remote supercomputer to a client PC. This calculation was done by connecting molecules in liquid phase to those in solid phase. After long iterations, we can see whether the material is stable in solid phase or not. Thus, we will see stability of the material at the target temperature.

V. Related Work

The Globus toolkit [13] provides a credential delegation based authentication mechanism which is embedded in its

Grid Security Infrastructure (GSI). GSI manages a mutual authentication between a local and remote computers as well as authorization on a remote computer. Security of the Globus builds on an X.509 user certificate. The certificate is used to create a temporary proxy certificate, which acts on behalf of user identification. CoG kit [14)] is available to build a portal site or a client grid application to utilize grid computing resources using the Globus toolkit. However a user should have sufficient expertise in a grid computing for installation and building stages. While the ITBL client API is available without installation process because it is an application-oriented library to be linked to a user's application.

UNICORE [15)] also provides a grid computing environment. Its authentication and authorization build on X.509 certificates, which are used to sign a job before submitting it to the UNICORE environment. An Arcon client library enables building client applications with a Java API.

Visualization in MD computations is one of the important issues. Many visualization software packages have been developed such as VMD [16)] and ProtoMol [17)]. VMD supports visualization for MD computations by using NAMD [16)]. NAMD is implemented in C++ and based on Charm++ parallel objects. Furthermore, its Tcl scripting provides analysis capabilities, allowing a user to design project-specific tools. ProtoMol is fully-equipped packages for MD computations from parallel computation to visualization. On the other hand, our client computing system orchestrates parallel computations on a remote supercomputer with visualization by using a native visualization program with the help of the ITBL's client library. Application user need not pay attention to the underlying middleware with the help of the ITBL's client library and a developed portable Java interface. Any kind of native visualization programs can be available independently with a grid computing environment.

VI. Conclusion

We have presented a client GUI computing system for securely accessing a remote supercomputer seamlessly by using the ITBL system. The GUI system has been redesigned to support seamless file manipulations and on-demand monitoring for standard output and standard error based on issues reported in our previous GUI system. Its infrastructure layer has been built by using an ITBL C++ client library. On top of the layer, we have deployed a user-friendly and flexible Java interface layer by using JNI. With the help of the ITBL's client library, we have realized single sign-on in user's authentication and authorization by using an X.509 certificate. Since the GUI system has adopted the same design in the previous one, it supports the same useful functionalities. Typically newly implemented seamless remote file manipulations remove gaps in handling files on a remote supercomputer. Monitoring support for standard output and standard error brings a chance to check progress of computations on demand. It is expected that this GUI system encourages nuclear material engineering by using a remote supercomputer.

Acknowledgment

We would like to thank Prof. Katsuyuki Kawamura, Tokyo Institute of Technology, Japan for providing us his MD program code named MXDORTOP and giving useful information. We would like to express our thanks to supporting staff of the ITBL system.

References

1) The International Framework for Nuclear Energy Cooperation (IFNEC), http://www.gneppartnership.org/
2) ITBL Project, http://www.itbl.jp/ [in Japanese]
3) SINET, http://www.sinet.ad.jp/?set_language-en
4) Y. Tsujita, T. Arima, K. Idemitsu, Y. Suzuki, H. Kimura, "Building an application-specific grid computing environment using ITBL for nuclear material engineering," *Proc. of ICONE16*, ICONE16-48223[CD-ROM] (2008).
5) Y. Suzuki, K. Sai, N. Matsumoto, O. Hazama, "Visualization system on information technology based laboratory," *IEEE Comp. Graph. Appl.*, **23**[2], 32-39 (2003).
6) Kawamura Laboratory, Tokyo Institute of Technology, http://www.geo.titech.ac.jp/lab/kawamura/eng/kawamuralab.e.html
7) MPI Forum, http://www.mpi-forum.org/
8) H. Inaba, R. Sagawa, H. Hayashi, K. Kawamura, "Molecular dynamics simulation of gadolinia-doped ceria," *Solid State Ionics*, **122**, 95-103 (1999).
9) T. Arima, S. Yamasaki, Y. Inagaki, K. Idemitsu, "Evaluation of thermal properties simulations from 300 to 2000 K," *J. Alloys Compounds*, **400**, 43-50 (2005).
10) H. Takemiya, N. Yamagishi, "Starpc: a library for communication among tools on a parallel computer cluster – User's and developer's guide to starpc –," JAERI-Data/Code 2000-06, JAEA (2000) [in Japanese].
11) VESTA, http://www.geocities.jp/kmo_mma/crystal/en/vesta.html
12) OpenMP, http://www.openmp.org/
13) I. Foster, C. Kesselman, S. Tuecke, "The anatomy of the grid: Enabling scalable virtual organizations," *Int. J. High Perf. Comp. Appl.*, **15**[3], 200-222 (2001).
14) G. von Laszewski, J. Gawor, P. Lane, N. Rehn, M. Russell, "Features of the Java commodity grid kit," *Conc. and Comp.: Prac. & Exp.*, **14**[13-15], 1045-1055 (2002).
15) D. Erwin, "UNICORE – a grid computing environment," *Conc. and Comp.: Prac. & Exp.*, **14**[13-15], 1395-1410 (2002).
16) J. C. Phillips, R. Braun, W. Wang, J. Gumbart, E. Tajkhorshid, E. Villa, C. Chipot, R. D. Skeel, L. Kale, K. Schulten, "Scalable molecular dynamics with NAMD," *J. Comp. Chem.*, **26**[16], 1781–1802 (2005).
17) T. Matthey, T. Cickovski, S. Hampton, A. Ko, M. Nyerges, T. Raeder, T. Slabach, J. A. Izaguirre, "ProtoMol, an object-oriented framework for prototyping novel algorithms for molecular dynamics," *ACM Trans. Math. Soft.*, **20**[3], 237–265 (2004).