

Experience and Lessons Learned from Enabling Applications for Users in National Grid Pilot Platform

G. Ong, B. Jiang

Singapore Computer Systems Ltd, 7 Bedok South Road,
Singapore 469272

gsong@scs.com.sg
jiangbb@scs.com.sg

Abstract

Availability of applications for end users has been identified as one critical success factor of National Grid Pilot Platform (NGPP) in Singapore. Therefore enabling applications for users has become one of the critical tasks of the National Grid Support Unit. In this paper we share much of the experience and lessons learned through this process. Areas covered include analysis and understanding of the applications handled by the team, and management of technical issues in terms of grid resource requirements and practical constraints in job execution.

Keyword: NGPP, Grid, Globus, Intel Compiler, MPICH-G2.

I. Introduction of NGPP and user support structure

The NGPP is the first phase of Singapore's cyberinfrastructure. This high speed network connecting technical computing resources has been conceived in response for a pragmatic approach to develop a cyber-infrastructure in Singapore. The NGPP aims to support Grid Computing for academia, research and industry. The stakeholders of the NGPP Phase 1 are the Agency for Science, Technology & Research (A*STAR), Defence Science & Technology Agency (DSTA), Economic Development Board (EDB), Infocomm Development Authority (IDA), National University of Singapore (NUS), Nanyang Technological University (NTU), and Singapore-MIT Alliance (SMA).

The initial participants (NUS, NTU, SMA and research institutes under the A*STAR) make available computer resources to support the NGPP. The NGPP is supported by the National Grid Support Unit (NGSU), staffed by technical team of Singapore Computer Systems Ltd. One of the missions of the NGSU is to support the NGPP end users to ensure that their applications are compatible and able to run over the NGPP resources. The NGSU also supports the operational aspects of the application execution.

The NGPP end users, who wish to have their applications enabled and/or executed over the NGPP resources, submit their project application to the National Grid Office (NGO), which coordinates and allocates the NGPP resources and support activities. Upon acceptance by the NGO, the NGSU will work with the end user concerned on the project for feasibility study, application analysis, development works, resource requirement analysis and execution planning.

II. Methodology adopted by the NGSU

The feasibility study forms the foundation of all NGSU project works. This is to ensure that the projects are feasible to achieve their objectives in the NGPP environment. The feasibility study revolves heavily around application analysis. The characteristics of the problem and the application is first analysed to ascertain whether or not the type of problem to be solved can benefit from the capacity of a Grid environment and whether or not the application design and code can be modified to achieve that objective. For instance, applications whose problem space can be broken down into smaller, independent work units are a good candidate for grid-enabling, whereas applications which are very tightly sequential in terms of execution and dataset processing will likely to have difficulty for grid-enabling or unlikely to benefit from a Grid environment.

We make references to the IBM Red Book (SG24-6936), Enabling Applications for Grid Computing with Globus. In essence, we analyse the following aspects of an application:

A. *Types of application flow*

Application flow is the flow of work between the jobs that make up the grid application. There are three types of application flows: parallel application flow, serial application flow, and networked application flow. The parallel application flow type is usually well suited for deployment on a grid, while more efforts may be required for the other flow types.

B. *Types of jobs*

Job is a single unit of work within a grid application. A job could utilize the services of the grid environment to launch one or more sub-jobs. It can be of any type: batch, standard application, parallel application, and interactive. The internal flow of work within a job itself is called job flow. Typically, parallel jobs are well suited to run in a grid environment whereas jobs of interactive nature can be difficult to benefit from grid.

C. *Practical factors*

We next consider various practical factors that will determine the eventual suitability of running the applications in a grid environment. The first consideration is the hardware requirements. An application may be architecture dependent. The number of CPUs and the storage size must be able to be satisfied by the executing grid environment. We also evaluate its software requirements, which can be in the form of dynamic libraries or toolkits and third-party software licenses.

Subsequently, a technical evaluation is made in terms of the compatibility of the programming language, availability of the relevant library code and the suitability of the overall execution environment for the application concerned. For instance, if the application is to be grid-enabled via MPI, we need to find out if there is a compatible MPICH-G2 library for linking with the application code.

Finally, the overhead of running in a Grid environment is to be benchmarked through test runs. This is to ascertain if the overhead will become too high to erase the benefits of the additional capacity in a Grid environment.

III. Applications projects handled by NGSU

Appendix A tabulates the applications projects handled by NGSU over a period of approximately 13 months. Out of the nine projects, P1 and P2 saw the most intensive works and many lessons learned.

They were the early batch of projects. P7, P8 and P9 turned out to be resource-only projects, where grid resources are required for execution of the applications concerned.

A quick analysis reveals some interesting facts. The majority of the projects (P1, P2, P3 and P4) to be grid enabled are MPI based. This makes the grid enabling works relatively easy by mapping MPI calls to MPICH-G2 calls. Linux is the most popular platform among the projects. (This is good news considering that Globus is primarily developed on the Linux platform.) Two major application areas among the projects are computational fluid dynamics and optimisation.

IV. Specific Lessons Learned

A. *Compile and linkage problem*

In the case of P1 and P2, the bulk of development time was spent on solving the compile and linkage issues. The application code is written for FORTRAN90. We use Intel FORTRAN90 compiler to build applications in NGPP. Ideally, we would simply compile the application code using the Intel compiler and link it against the MPICH-G2 library, which is pre-built with GNU gcc compiler. However, this approach produced many errors during the linkage stage due to the different ways the two compilers handle object symbols. The logical option is to rebuild MPICH-G2 library with the Intel compiler. However, this is not practical as the library relies heavily on Globus libraries, which are built with GNU gcc compiler.

The question is therefore whether or not it is possible to use the Intel FORTRAN compiler to link a program against a library that has been compiled with GNU FORTRAN compiler. There are primarily two major issues:

1. default underscoring convention of GNU library is incompatible with Intel compiler;
2. libg2c used by GNU FORTRAN compiler and the EPC library used by Intel FORTRAN compiler are incompatible.

GNU FORTRAN compiler has an option to omit the second appended underscore in the linker symbols – “-fno-second-underscore” flag – which makes the .o files it produced compatible with INTEL FORTRAN library. This overcomes the first issue.

We overcame the second issue by recompiling one specific FORTRAN source file in MPICH-G2 using the Intel FORTRAN compiler. “farg.f” contains references to the functions iargc() and getarg(), which produce linkage errors when the GNU FORTRAN compiler-generated version is used. We recompiled the code using the Intel FORTRAN compiler and used “ar” utility to replace the generated object code in “libmpichg2.a”. This generated a “compatible” version of the library that can link without errors.

B. *Resource accessibility issue after converting from MPICH to MPICH-G2*

To run the MPI-based applications on the grid, MPICH-G2 must be used on the grid resources. MPICH-G2 requires TCP communication between the nodes where the application processes are scheduled and hence it does not work with clusters with compute nodes in private address space. This characteristic makes many of the compute nodes on NGPP unsuitable for execution of the applications.

Realm Specific Internet Protocol (RSIP), a solution developed by Hewlett Packard, in conjunction with DNS service can overcome this problem. To connect a private network to a public one, an RSIP gateway should be put on the boundary between networks. It owns a pool of public IP addresses that it can allocate to hosts on its private network. A pilot test run of RSIP on some NGPP resources has proven that it could indeed overcome the shortcoming of private address space on resource availability. At the time of writing, RSIP is being deployed on the NGPP resources. A benchmark test is being planned as performance bottleneck is perceived to be a potential issue on such architecture.

C. Support scalability

While working on P1, the NGSU team was confronted with repeated updates to the code that was being worked on. Apparently the owner of the code was still testing and enhancing the code while it was being grid-enabled. This posed a challenge to the porting effort in keeping up with the new code. It led to the idea that the owner himself should be made responsible for the porting work. Furthermore, more application projects have been received and the trend indicated that the NGSU support capability and effectiveness would be challenged.

The idea essentially demands a programme to empower the application owners to help themselves with the grid enabling works. The National Grid Competency Centre has already developed and conducted a number of introductory courses on grid computing and Globus. These provide solid foundation to the aspiring application owners. In addition, the NGSU team has also developed a specialised course “GEU-2: Grid-enabling of applications” to train the application owners with the necessary skills and hands-on experience.

In addition to the training framework above, the NGSU team has also set up a test grid environment to facilitate testing of the applications. Customised build and testing guides are also developed to some of the applications as part of the hand-over process in the development works.

V. Conclusion

The project works carried out over the 13 month period provided us tremendous experience and exposure to the real problems that one would face in grid enabling applications. The incremental experience in this relatively new field is critical for better understanding of the technical issues and more effective support of user requirements. We believe that, given the choice, a richer pool of applications from different domains will further consolidate our understanding and appreciation of the challenges faced in grid enabling works.

Appendix A: Application projects handled by NGSU as at Dec 2004

Project Name	Application/ Domain	Platform/ OS	Hardware/ architecture requirements	Resource Requirement (No. of CPU, Hard disk)	MPI based?	Compiler	Has the code been tested in cluster	Estimated Efforts Required (man-days)	Status as at Dec 2004
Incompressible Turbulent Flows	CFD	Linux	N/A	28 CPUs, 6 GB	Yes	F90	Yes	15	Code enabled and execution in progress
Numerical Simulation of Flow over Dimpled surface	CFD	Linux	N/A	512 CPUs , 20 GB	Yes	Intel F90	Yes and further development to be integrated	15	Code enabled, execution pending resource availability
A Cartesian Based Euler Method for the Computation of Complex Configurations	CFD	Linux	N/A	32 CPUs , 15 GB	Yes	F90	Yes	20	Code enabled, ready for execution
A Framework for Multi-Disciplinary Optimization	Optimization	Linux	N/A	50-100 CPUs, 5 GB	Yes	gcc	Yes	20	
Flexible Concurrent Global Optimization	Optimization	Linux	Intel	>= 4 CPUs, 2 GB	No	gcc	No	15	Enabling works in progress
Development of Novel Grid-enabled Evolutionary Computation Algorithms to Solve Large-scale Complex Optimization Problems in Grid Computing Environment	Optimization	UNIX	>800MHz Itanium	20 CPUs, 20 GB	No	Matlab	Yes	20	Project KIV
Grid-based Data-mining Algorithm using Subset Trained	Data mining	UNIX	N/A	32 CPUs, 10GB, max. of	N/A	N/A	N/A	N/A	Resource requirement only. No enabling

	Support Vector Machines				2GB per resource					works.
	Multiscale Modeling of Nanomaterials and Nanostructures by Grid Computing	Nanotech	UNIX	N/A	32 CPUs, 50GB	N/A	N/A	N/A	N/A	Resource requirement only. No enabling works.
	Collaborative Problem Solving Environment for Complex Engineering Designs using Grid Computing Technology	Engineering Design	Linux, IBM AIX	Intel/Opteron, IBM Power4	16-32 CPUs, 50GB	N/A	N/A	N/A	N/A	Resource requirement only. No enabling works.

Abbreviations used:

CFD – Computational Fluid Dynamics



Ong Guan Sin is currently Grid Manager of SCS, responsible to develop its grid business from scratch since 2003. He built his first grid system in 1997 when he participated in RC5 cracking effort on the Internet. Prior to joining SCS, he was with Singapore Management University as Senior Operations Manager, building the university's infrastructure and operations from a small setup to a highly available environment today. Earlier, he spent his entrepreneurial years building multilingual Internet addressing technologies and e-commerce ventures.



Jiang Bei Bei is a Systems Specialist from SCS. She works on porting and benchmarking of the parallel applications in the Grid environment. Her interest is mainly on the methodology and the underlying mathematical structures, with a special focus on their connections to the computational procedures. Before joining the company, she obtained her M.Sc. degree in High Performance Computation for Engineered Systems from the Singapore-MIT Alliance.