Development High Performance Scientific Computing Application Using Model-Driven Architecture

Bazargul Matkerim

Mechanical-Mathematical faculty
Al-Farabi Kazakh National University
Almaty, Kazakhstan
bazargulmm@gmail.com

Darhan Akhmed-Zaki

Mechanical-Mathematical faculty
Al-Farabi Kazakh National University
Almaty, Kazakhstan
darhan.ahmed-zaki@kaznu.kz

Manuel Barata

Department of Electronics, Telecommunications and Computer Engineering
Lisbon Superior Engineering Institute
Lisbon, Portugal
mmb@deetc.isel.pt

Copyright © 2013 Bazargul Matkerim, Darhan Akhmed-Zaki and Manuel Barata. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.
Abstract

This paper presents an approach to develop high performance scientific computing (HPSC) application for simulation fluid flow pressure in oil reservoir with use of mpiJava. The modeling and developing process adapted the modern Model-Driven Architecture (MDA) technology which was created by Object Management Group. We specially created the HPSC components to model the platform independent model (PIM) and platform specific model (PSM) of MDA. The HPSC application development was analyzed with traditional and MDA approaches.

Keywords: HPSC, oil reservoir simulation, mpiJava, Model-Driven Architecture

1 Introduction

Oil reservoir simulation is one of the scientific computing research domains which require more computing resources. Nowadays with the development of high performance computing system it is advisable to take advantage of them by developing the corresponding programs and obtain computing performances much faster than ever before.

Traditional scientific computing process of oil reservoir simulation consists of several phases namely: the physical modeling of oil reservoir, mathematical modeling of the physical phenomenon, numerical modeling the mathematical model, coding the mathematical model in programming language and validation and verification testing as shown in fig. 1-a. It is well known that, in this approach, the developing process is often driven by the mathematical model, which is described in a mathematical language. The drawbacks of traditional HPSC application such as architecture dependency, mix-up of concerns and programming complexity, as mentioned in [8], causes the use of parallel programming models at low level, machine specific and complicate application porting.

We aimed to use MDA technology for HPSC in oil reservoir to get rid of the drawbacks of traditional application development. We choose the mpiJava [4, 12] which is an object-oriented Java interface to the standard Message Passing Interface (MPI) [10], because we want to design the source code using the Java programming language. We believe the modern way of software engineering will help us to develop extensible, flexible and maintainable HPSC applications.
2 Problem statement

Our target is to simulate the fluid dynamics in the reservoir for recovery the oil and gas. The pressure is one of the main research objects like permeability, saturation. The water or gas injection wells always provide pressure supports to
production wells [3], since it is costly to drill and operate the wells, the application of simulation tools play an important role.

2.1 Mathematical model of fluid flow pressure in oil reservoir

The fluid flow in porous elastic anisotropic medium in the cube \( \Omega = [0, T] \times K \{0 \leq x \leq 1, 0 \leq y \leq 1, 0 \leq z \leq 1\} \) is described by the following set of 3 dimensional differential equations:

\[
\frac{\partial P}{\partial t} = \frac{\partial}{\partial x} (\phi(x, y, z) \frac{\partial P}{\partial x}) + \frac{\partial}{\partial y} (\phi(x, y, z) \frac{\partial P}{\partial y}) + \frac{\partial}{\partial z} (\phi(x, y, z) \frac{\partial P}{\partial z}) + f(t, x, y, z) \quad (1)
\]

with initial and boundary conditions:

\[
u(0, x, y, z) = \phi(0, x, y, z) \quad (2)
\]

\[
\frac{\partial P}{\partial n} |_{\Gamma} = 0 \quad (3)
\]

Here in (3) \( \Gamma \) is the surface area of \( \Omega \) cube.

In equation (1), the solution function \( P(t, x, y, z) \) is reservoir pressure at point \( (x, y, z) \) at time \( t \); \( \phi(x, y, z) \) is diffusivity coefficient of the reservoir \( f(t, x, y, z) \) is density of the sources, the deposit of wells. We used implicit numerical method Jacobi to solve the problem of (1) - (3).

2.2 Develop HPSC application in MDA

The popular and promising initiative of Model Driven engineering (MDE) is the Model-Driven Architecture (MDA) of Object Management Group (OMG). MDA is successfully applied in many application domains [9]. There are scientists who are beginning to use MDA in scientific computing in Molecular Beam Epitaxy, reflection High-Energy Electron Diffraction [1] and CEA scientific applications [3].

As can be seen in fig. 2 according the MDA technology we could see three viewpoints for the HPSC application: computational independent model (CIM), platform independent model (PIM) and platform specific model (PSM). In HPSC
Development high performance scientific computing application

domain the CIM is a mathematical model which is a mathematical representation of an application from the computation-independent viewpoint. The mathematical modeler need not to have knowledge about the models or artifacts used to realize the functionality for which the requirements in mathematical model are articulated in the CIM. The PIM is transformed from the CIM, which is independent of the following [6]: the third-generation and fourth-generation languages, such as Java, C++, C#, and Visual Basic; distributed component middleware, such as J2EE, CORBA, and .NET. The PIM viewpoint of HPSC application is technology-neutral which define the operation of a system while hiding the details necessary for a particular platform. The PIM could be transformed into one or more than one required PSM depend on the particular requirements of HPSC application. In current practice the HPSC applications are mostly in CUDA, MPI, OpenMP and MapReduce platform technology or the hybrid of two or three of them depend on the HPSC hardware resources, and the coding is in Java or C/C++ languages. At last phase the PSM is transformed to corresponding java or C/C++ code in required specific technology automatically.

In fig.1- a and b schematically presented the scientific computing software development process in traditional way and MDA respectively with UML activity diagram. As shown in these figures the modeling runs on the whole process of scientific computing. But the difference of them is in fig.1-a between the numerical modeler and programmer is added the model-driven development modeler. The modeler is who bridging the gap between mathematicians and programmer with PIM and PSM.
Bazargul Matkerim, Darhan Akhmed-Zaki and Manuel Barata

Fig. 2. HPSC application model transformation according MDA technology

In traditional way, scientists code the application in C or Fortran languages manually, guided by the numerical model. But recently, due to the complexity of the platform, high performance system, mathematical and numerical algorithm, it is not anymore practical.

2.3 Technical solutions of MDA for HPSC

For HPSC in oil reservoir the technologies suggested in [5] are adoptable. We have chosen the most remarkable Integrated Development Environment (IDE) of Eclipse [7] and performed the MDA technology on it. It is possible to add various plug-in to Eclipse which support the MDA.

As we described above, in MDA technology, the computational development process of HPSC (1) – (3) problem application in mpiJava consists of three major steps as following:

1. Create PIM with object-oriented UML 2.0 class diagram.
2. Transform PIM to PSM.
3. Transform the mpiJava PSM to code with UML to code generator.

So let’s consider the realization of first step. As can be seen in Fig.1-b at the fourth phase stays the PIM which represents the numerical model into high-level computational model. We abstract out 10 common components from HPSC
application in order to build PIM with UML class diagram. Here the components are the HPSC elements with which could build up the various software applications. Each HPSC component class is defined with different properties and operations depending on its functions in computation. The HPSC components are listed below:

1. CartCreate: get the rank and size of the given available processors and create one, two or three dimensional Cartesian topology for computing.
2. DefComDomain: define computing domain in length of x, y and z and the time.
3. ComBlock: decomposition of the defined computing domain into small computing blocks.
4. ProcCom: define the boundary data communication between processors in topology by the location in given computing domain.
5. MathMesh: define the mathematical mesh in type of structured, unstructured or hybrid for computing.
6. EquationCon: define the initial and boundary conditions of HPSC problem.
7. NumericalMethods: collections of the implicit or explicit numerical methods.
8. Equations: define the type of equations by numerical modeling.
9. WriteToFile: write the result to file.
10. Main: main class to define the given equation which should be computed.

The PIM is shown in fig.4-a. We divided the defined components into four groups by their functions:

- The first group of components are CartCreate, DefComDomain, ComBlock and ProcCom, is used to formalize the common HPSC technologies such as MPI, OpenMP, CUDA or MapReduce;
- Second group of components are MathMesh and NumericalMethods, is used to define the numerical model with various type of numerical grids and methods;
- Third group of components are Equations and EquationCon, is used to define the physical-mathematical model with numerical finite differentia equations and conditions for these equations;
- Fourth group contains the Main and writeToFile components with which we could organize the HPSC application input and output.

So with the four groups 10 component classes we could define various equations, conditions, numerical methods and mapping to the HPSC hardware.

On the second step, due to our chosen platform is Java and HPSC technology
is MPI, so we transform the PIM to mpiJava PSM. As can be seen in fig.4-b each UML component class is transformed into a Java class which satisfies the mentioned specific platform.

For realization of the last step we used free and open source UML to Java generator [16] of Obeo [14] which is based on the Accello [13]. It allows us to produce Java source code from mpiJava PSM. Since the generator is customizable we could contribute dynamically to modify our generated Java code. Fig.5. shows the fragment of code of MathMesh component. The transformation to code means: get the Java classes in text form of code from model elements such as classes, fields, methods, interfaces, and enumerations. It is noteworthy that we get from the transformation in other words is the skeleton of 10 components Java classes and the implementation of methods filled by ourselves.

We presented the difference between HPSC java application in traditional way and in MDA technology as shown in Fig.6-a, b. We know that the development of HPSC application in traditional way is driven by the physical-mathematical model which is manually coded directly from the chosen numerical model. Here the main problem is the lack of separation between physical-mathematical and HPSC technology concerns. Of course, it is difficult to retrieve afterwards the physical models from the source codes with mixed concern. In this case the complexity of maintenance and evolution developed applications can be imagined.

![Fig.4.a. Screenshot of PIM model of HPSC application b. Screenshot of PSM model of HPSC application from Eclipse interface.](image-url)
The development based on Model-Driven development (MDD) in MDA results in a much easier way to modify and extend HPSC applications than the traditional way because of division of concerns with the four group of components. Each group of component classes is extendible with adding required functions with Java methods. For example in NumericalMethods class we have defined common implicit and explicit methods, so in the Main class we could implement these methods, but if we want to use other numerical method which wasn’t included in the NumericalMethod then we add this method in PIM level and transform it to PSM and then to Code. The way of adding other available computing methods to NumericalMethods class is carried out on the conditions of without affecting existing application. By using MDA technology it is possible to change HPSC technology of one scientific problem PIM by transform it to other HPSC PSM, and is possible to modify existing PIM for other scientific problem with the help of component methods. Each component is easy to be modified to a wide variety of numerical computing model. So we could model these elements once, and use them again and again when we need for various scientific problems.

Fig.5. The Mathematical mesh definition class “MathMesh” which is automatically generated by PSM to Code transformation
3 Results of computation

The complete Java programs for problem (1)-(3) in MDA and in traditional method are tested on URSA [14] with 8, 16 and 48 cores with the simulation nodes of 64x64x64, 128x128x128 and 240x240x240. The HPSC mpiJava applications which were developed in traditional way and in MDA technology give the same computing results as sequential java application of the problem. The parallel computing speedup and efficiency are shown in fig.7 and fig.8 separately. Here in fig.7 index of S 1, 2 and 3 means the three types of grid nodes and after the underscore 1, 2 means the traditional program computation and MDA program correspondingly. From fig. 7 we could see the speedup of MDA version is higher than the traditional one and more efficient except the S1_1 and S2_2 at the core of 48.
Fig. 7. HPSC speedup in mpiJava

Fig. 8. HPSC efficiency in mpiJava
4 Conclusions

This paper presented an approach of development HPSC applications with MDA technology for pressure in fluid flow in oil reservoirs. Generally speaking from this study we believe that it is possible to model and develop HPSC application more flexibly with MDA than traditional way. We find MDA technology is benefit for develop HPSC application by the advantages as following: (I) the HPSC application for various scientific problem could be modeled with four object-oriented computing component groups in PIM with UML2.0 (II) the high level PIM could be transformed to low level HPSC technology PSM and transformed the PSM to chosen source code automatically (III) The defined four group components is about 80% stable for when modification to other scientific computing problem, so the maintenance and extension solved easily with the PIM and PSM. (IV) The heavy problem of documentation the development process becomes simple because of visual model of PIM and PSM.

We only presented the HPSC fluid flow pressure using MDA technology, further computing set of equations of fluid flow dynamics in oil reservoir for saturation, speed, temperature and concentration will be summarized in our next study.
**References**


[16] UML to Java generator:
http://marketplace.obeonetwork.com/module/uml2java-generator


Received: July 25, 2013