Concurrent Engineering of an Object-Oriented Particle-In-Cell Simulations Environment for Physicists

Fatma Dandashi, James Acquah, Sudhanshu Killedar,
Hadon Nash, William Pritchette, David Rine
Computer Science Department, George Mason University,
Fairfax, VA 22030 email: dandashi@cs.gmu.edu *

Abstract- This paper reports on an ongoing research effort that is being jointly conducted by faculty and students at the CS Department at George Mason University, in Fairfax, Virginia, and by faculty and students at the EECS Department at UC Berkeley, California. The research is aimed at developing an Object-Oriented modeling environment for Particle In Cell plasma simulations.

1 Introduction

One of the major areas of computational physics is computational plasma physics. A fundamental approach in developing plasma simulations is the Particle-In-Cell (PIC) approach whereby PIC modeling and simulations is an important, but so far costly, first step in developing numerous devices. Software performing PIC simulations is well known among physicists working in that domain [1]. These simulations, written in procedural languages such as FORTRAN, are tedious to use, being hundreds of thousands to millions of lines of code, and very difficult to modernize and extend. To make available to a large number of science users, and to modernize and extend current PIC simulation capabilities to include a more maintainable Graphical User Interface (GUI) and a more maintainable PIC content library, an Object-Oriented PIC (OOPIC) simulation development environment is being developed in C++ to run under MS-Windows.

2 OOPIC Development Process

Before the OOPIC simulations development environment could be developed, a PIC [5] domain model had to be constructed. In the early requirements phase of OOPIC development, software developers were faced source codes and converted into an OO domain model. Alternately, because PIC domain expertise was readily available and earlier PIC software codes were too diverse, a new domain model could be developed using current domain experts. It was decided that the knowledge of experts in the problem area should be utilized to develop and validate a new domain model. Using this approach, the result is usually an accurate representation of the structure of the domain. Added confidence is gained in our case because the domain experts are also end users of the derived PIC simulations and potential re-users of the PIC C++ classes that have been developed for the PIC content library. In constructing the domain model, the researchers divided the OOPIC application into two major subdomains, PIC and GUI. Two reasons supported the decision for this concurrent development [4] of PIC and GUI, together comprising approximately two million lines of source code.

- First the modules for these two subdomains were functionally independent of each other. The GUI subdomain is part of the technology domain, while the PIC subdomain is part of the applications domain. The GUI is inherently a generic application. It may initially be developed independently from the application software using it. On the other hand, PIC software is application dependent. While it is being designed for subsequent reuse by interested physicists, it is ultimately physics intensive and functionally separate from any GUI application.

- The second reason for the separation of these two subdomains was a logistic one, necessitating concurrent engineering [4] in a geographically distributed environment. Supported by the independence of the two subdomains, two geographically separate teams are responsible for the software development. A team of physicists, located in California, is responsible for developing the PIC code components [7]. A team of software engineers, located in Virginia,

is responsible for developing the GUI code components [3]. Networking services support this concurrent engineering activity. In addition workshops are held semi-annually to facilitate communication and the direct exchange of information and development strategy.

3 Development Concerns

However, during development, some sharing and synchronization of work in developing the PIC and GUI subdomain modules was unavoidable. This has led to some delays in meeting certain initial project schedule deadlines because of obvious geographical obstacles that impeded more knowledge-rich communications. Another reason for some delays was the emerging realization that the two teams differed on programming design philosophies. The main interest of the PIC team was to quickly deliver an environment that allowed advanced simulations at the expense of some quality factors. The GUI team, on the other hand, was more concerned with producing generic, well designed, well documented, easy to maintain and extend software at the expense of early performance factors. These two differing points of view have led to some early tension and final production delays, while the advantages and disadvantages of the two approaches were being negotiated. We feel that these exchanges have helped the computational physicists understand and appreciate the care and time that must be spent on software development to produce quality code, and to understand the need to spend more initial time on risk analyses [2]. Similarly, the software engineers have emerged with a new understanding of what is involved in working with software developers whose main area of expertise is a complicated problem domain. Another source of concern that has emerged during OOPIC development has been in the technology subdomain of the simulations development environment. During the initial development stage, and after careful market assessment, a decision was made to build OOPIC on a certain software platform. The decision was based on the state of the software market at the time. Since then, the chosen vendor has failed to deliver some of the capabilities and the reliability required of a software platform to ultimately support OOPIC. Another software vendor has since then offered a series of more advanced software building tools, such as reliable compilers for C++, and advanced class library tools. Consequently, it now seems that the project development effort would have been better served, time wise, with a different vendor choice. Future enhancement efforts may include a technology transfer to another software platform. An ultimate solution to this problem is to develop a more flexible technology subdomain modeling method affording an efficient and effective migration to new supporting technologies that become available. Another GMU graduate student has partially solved this problem [6].

4 Potential Benefits

Existing PIC codes have traditionally been developed for mainframes and supercomputers. OOPIC will now allow interested computational physicists everywhere to run advanced PIC simulations without requiring the very expensive, complex computing facilities that traditional PIC codes have demanded. Furthermore, by virtue of its OO design, the advances in chip technology, and the plethora of software products that exist today, OOPIC will allow physicists to tackle new problems and will make the composition of new simulations from the OOPIC code modeling and simulation environment easier. According to one plasma physicist working on the OOPIC project, the new code is much easier to modify and tailor than any other PIC code that exists in the public domain today. The Beta release of the code is targeted for January 1995.

References

[1] Birdsall, C.K. and A.B. Langdon, *Plasma Physics Via Computer Simulation*, McGraw-Hill, New York, 1985, Reprinted by Adam-Hilger, 1991.

[2] Boehm, B.W., "A Spiral Model Of Software Development and Enhancement," *ACM Software Engineering Notes*, Vol. 11, No., 4, Aug. 1986, pp. 14-24.

[3] Dandashi, F., W. Pritchette, and D.C. Rine, "A Metric Suite for ++ Class Libraries," *Proceedings of the TOOLS 14 Conference*, Santa-Barbara, CA, Aug. 1994, pp. 499-508.

[4] Ramana Reddy, Y.V., K. Srinivas, V. Jagannathan, and R. Karinthi, "Computer Support for Concurrent Engineering," *IEEE Computer*, Vol. 26, No. 1, Jan. 1993, pp. 12-16.

[5] Rumbaugh, J., M. Blaha, W. Premerlani, F. Eddy, and W. Lorensen, *Object-Oriented Modeling and Design*, Prentice-Hall, Englewood Cliffs, NJ, 1991.

[6] Ta, A., "Technology-Based Domain Analysis and Modeling Method," Technical Report, CS Department, SITE, George Mason University, Sept. 1994.

[7] Verboncoeur, J.P., and N.T. Gladd, "Application of Object-Oriented Design to Particle-In-Cell Plasma Simulations," *Proceedings of the TOOLS 14 Conference*, Santa-Barbara, CA, Aug. 1994, pp. 341-352.