A Modular Collaborative Parallel CFD Workbench

Kwai L. Wong[1] and A. Jerry Baker[2]

[1] Joint Institute for Computational Science, University of Tennessee, Knoxville, TN 37996, USA
[2] Engineering Science Program, MAE&ES, University of Tennessee, Knoxville, TN 37996, USA

Abstract. Simulation of physical phenomena on computers has joined engineering mechanics theory and laboratory experimentation as the third method of engineering analysis design. It is in fact the only feasible method for analyzing many critically important phenomena, e.g., quenching, heat treating, full scale phenomena, etc. With the rapid maturation of inexpensive parallel computing technology, and high performance communications, there will emerge shortly a totally new, highly interactive computing/simulation environment supporting engineering design optimization. This environment will exist on Internet employing interoperable software/hardware infrastructures emergent today. A key element of this will involve development of computational software enabling utilization of all HPCC advances. This paper introduces the concept and details the development of a user-adapted computational simulation software platform prototype on the Internet.

1 Introduction

Simulation of physical phenomena on computers has joined engineering experiment and theory as the third method of scientific investigation. It is in fact the only feasible method for analyzing many different types of critically important phenomena, e.g., geological time scale evolution, options for clean/efficient combustion, quenching/heat treating, optimized ventilation design, etc.

The need to address such problems involves the collaborative work of scientists, engineers, and computer specialists. With the rapid infusion of parallel computing technology in recent years, the opportunity to create a unified problem-solving environment in computational mechanics to attack large-scale problems has emerged.

To meet the challenge, many issues have to be thought out and resolved. Foremost, an interoperable software infrastructure has to be built, which deals with issues like physics encapsulation, I/O streaming, heterogeneity, security, etc. Last fall, a team of engineering, computational and software scientists, at the University of Tennessee/Knoxville (UTK) started a collaborative inhouse research project to address the theoretical, algorithmic, and numerical implementation issues necessary to assess the potential for success of a Parallel Interoperable Computational Mechanics Simulation System (PICMSS) operating on

V.N. Alexandrov et al. (Eds.): ICCS 2001, LNCS 2073, pp. 336–344, 2001.

the Internet. This theoretical focus is development of an implicit finite element multi-dimensional CFD software platform capable of admitting diverse CFD formulations, as well as closure models for physics phenomena, as required/derived by users.

Imagine that engineers in industry and/or academic researchers will be enabled to:

1. Remotely describe their problem to a genuinely production simulation environment from their local PC,
2. Submit their computational tasks to a pool of collaborative computers located anywhere in the country,
3. Monitor their computational tasks interactively, then,
4. Examine their final results locally from remotely available graphics utilities.

This is exactly the environment that PICMSS aims to provide. To deliver such capabilities, this project will integrate innovative software design, computational mechanics theory and parallel CS packages with proven methodologies [1,4].

2 Objective

The objective is to develop, advance and proliferate a Parallel Interoperable Computational Mechanics Simulation System (PICMSS). The common lament in the engineering computational mechanics community is the lack of adaptability in available software for effectively and efficiently attacking multi-disciplinary problems in engineering design.

In constructing PICMSS, the goal is to collate the significant recent advances in the communication and information sciences, and in computational mechanics theory/practice, and to deploy the result into the hands of both researchers and engineering practitioners to better facilitate rapid simulation of engineered systems involving non-linearly coupled fluid/thermal mechanics disciplines. PICMSS will as well be designed as a faculty toolkit to support academic instruction/research in the computational mechanics of continua with operation/dissemination via Internet.

The specific goal is to develop and deliver as a national/international resource:

1. A modular parallel compute engine using, but not limited to, the finite element (FE) implementation of a weak form to computationally formulate the physics, chemistry and mathematics description of the problem statement at hand,
2. An equation-based graphical client user interface to encapsulate the FE computational theory for this differential/algebraic equation system describing the multi-disciplinary problem to be simulated,
3. A middle server layer adapted to NetSolve [2] to steer and monitor the computational state of the defined simulation, and
4. Enhancements to NetSolve to facilitate data exchange between the computing engine and users or secondary storage brokers during and after execution.

3 Scope of PICMSS

The simulator will be assembled at six levels of abstraction:

1. Construction of a JAVA graphical user interface (GUI) uniquely tailored to suit the implementation of the problem framework using language constructions familiar to engineers and researchers,
2. Conversion of this differential/closure equation system into a computational form using state-of-the-art numerics,
3. Creation of backend server interface to control and distribute computing information to processing units,
4. Expansion of the computational engine to admit various classes of problems,
5. Induction of PICMSS to NetSolve for metacomputing,
6. Exposition of output data and monitoring and control during execution.

Figure 1 graphs schematically the three-tier structural units of the PICMSS computational simulator. A client-server model is used to provide the interfaces between the users and the meta-computing engines. The PICMSS translation from GUI differential equation system to computational form will be based on, but not limited to, a finite element discrete implementation of a weak form. An equation editor will be developed for users to express their problem statements, constituted of non-linearly coupled partial differential, algebraic and algebraic-differential equations.

The language will be vector calculus, using the "textbook" form familiar to engineers. Geometrical (definition, discretization, etc.) inputs and ini-

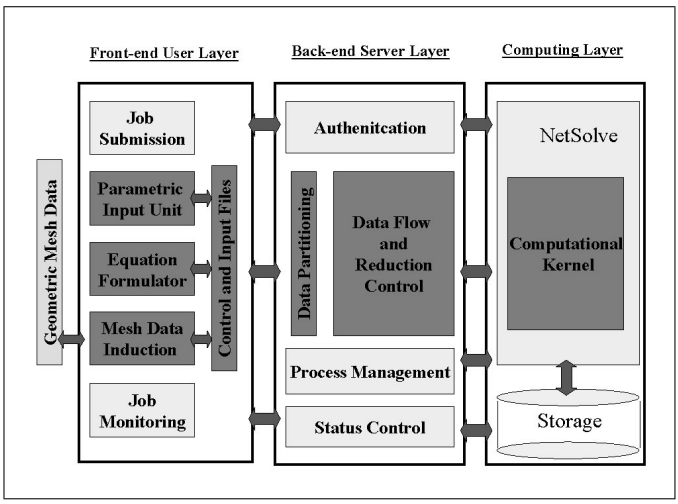

Fig. 1. Three-Tier Structural Units of PICMSS.

tial/boundary conditions will be admitted to PICMSS as external files created

at the client site. The encapsulated information will pass through a JAVA server deamon to manage and schedule resources for the computational engine to act on. The computational kernel of the simulator can be located independently on a pool of local machines, or be put under the control of NetSolve for global access. Output is piped to designated resources for local retrieval or remote examination.

The Front-end Client Interface will allow users to define, formulate and monitor their problems. The back-end Java server is designed to provide basic services for security, resource location, resource management, status monitoring, and data movement that are required for high performance computing in distributive environments. The computational kernel accepts the input file and executes the sequence of tasks defined in the control file. It is built on modular differential and integral operators acting on operands (state variables).

The encapsulation of mesh construction and assembly procedures increases the modularity and portability of the method to attack problems of different nature and formulations. It is also inherently parallel. The "solving" of partial differential equations generally involves a time integration scheme, a Newton or quasi-Newton iteration strategy for nonlinear corrections, and a linear solver for a large scale matrix statements. The use of Krylov iterative methods [1,4] is primarily coupled with preconditioners to improve convergence as the problem size increases.

To insure that PICMSS is well integrated into the Computational Grid movement that is now emerging, it will utilize NetSolve, which has emerged as a leading software environment for building grid-enabled PSE's [3]. NetSolve (www.cs.utk.edu/netsolve/) is a software environment for networked computing that transforms disparate computers and software libraries into a unified, easy-to-access computational service.

4 Benchmarking PICMSS

Benchmark problems to verify the integrity, validity, and scalabilty of PICMSS include three-dimensional flows in a straight channel, a lid-driven cavity, and a thermally-driven cavity. Exact solutions for the velocity profiles of the fully developed duct flow are available in two and three dimensions [5]. Data of computational benchmarks are also available for the three dimensional driven and thermal cavities. Using the velocity-vorticity formulation for the incompressible Navier Stokes equations, a collection of solutions [6] for a range of different Reynolds numbers and Rayleigh numbers were developed under a predecessor effort on various parallel machines. Herein, the computed results for benchmark 2D and 3D duct flow are presented using both the velocity-vorticity incompressible Navier-Stokes (INS) formulation and the pressure projection INS formulation.

4.1 Governing Equations

The nondimensional, laminar flow incompressible Navier-Stokes equation system with Boussinesq body-force approximation in primitive form is,

Continuity:

$$\nabla \cdot \mathbf{u} = 0, \tag{1}$$

Momentum:

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} = -\nabla P + \frac{1}{\mathrm{Re}}\nabla^2 \mathbf{u} - \frac{\mathrm{Gr}}{\mathrm{Re}^2}\Theta \hat{g}, \tag{2}$$

Energy:

$$\frac{\partial \Theta}{\partial t} + (\mathbf{u} \cdot \nabla)\Theta = \frac{1}{\mathrm{RePr}}\nabla^2 \Theta, \tag{3}$$

where $\mathbf{u} = \mathbf{u}(\mathbf{x}, t) = (u, v, w)$ is the velocity vector field (resolution), t is the time, $\mathbf{x} = (x, y, z)$ is the spatial coordinate, Θ is the potential temperature, $\hat{\mathbf{g}}$ is the gravity unit vector and P is the kinematic pressure. The nondimensional parameters are Reynolds number (Re), Prandtl number (Pr), and Grashof number (Gr) defined as,

$$Re = \frac{U_r L_r}{\nu} \quad , \quad Pr = \frac{\nu}{\alpha} \quad , \quad Gr = \frac{\beta g \Delta T_r L_r^3}{\nu^2},$$

where L_r and U_r are the reference length and velocity respectively, g is the gravity acceleration, ν is the kinematic viscosity, β is the coefficient of volume expansion, α is the thermal diffusivity, and ΔT_r is the reference temperature difference. The vorticity vector, $\mathbf{\Omega} = (\Omega_x, \Omega_y, \Omega_z)$, is kinematically defined as

$$\mathbf{\Omega} = \nabla \times \mathbf{u}. \tag{4}$$

Taking the curl of definition (4), together with the incompressibility constraint (1) and the vector identity,

$$\nabla \times \mathbf{\Omega} = \nabla \times \nabla \times \mathbf{u} = \nabla(\nabla \cdot \mathbf{u}) - \nabla^2 \mathbf{u}, \tag{5}$$

yields the velocity vector Poisson equation system

$$\nabla^2 \mathbf{u} = -\nabla \times \mathbf{\Omega}. \tag{6}$$

Taking the curl of the momentum equation (2) eliminates any gradient field. Applying equation (1) and noting $\nabla \cdot \mathbf{\Omega} = 0$, the vorticity transport equation is

$$\frac{\partial \mathbf{\Omega}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{\Omega} - (\mathbf{\Omega} \cdot \nabla)\mathbf{u} = \frac{1}{\mathrm{Re}}\nabla^2 \mathbf{\Omega} - \frac{\mathrm{Gr}}{\mathrm{Re}^2}\nabla \times \Theta \hat{\mathbf{g}}. \tag{7}$$

Hence, the velocity-vorticity formulation for the laminar INS equations system with Boussinesq approximation in three dimensions can be written as

$$\nabla^2 \mathbf{u} + \nabla \times \mathbf{\Omega} = 0, \tag{8}$$

$$\frac{\partial \mathbf{\Omega}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{\Omega} - (\mathbf{\Omega} \cdot \nabla)\mathbf{u} - \frac{1}{\mathrm{Re}}\nabla^2 \mathbf{\Omega} + \frac{\mathrm{Gr}}{\mathrm{Re}^2}\nabla \times \Theta \hat{\mathbf{g}} = 0, \tag{9}$$

$$\frac{\partial \Theta}{\partial t} + (\mathbf{u} \cdot \nabla)\Theta - \frac{1}{\mathrm{RePr}}\nabla^2 \Theta = 0. \tag{10}$$

This formulation consists of three velocity Poisson equations that couple the velocity and vorticity components kinematically via the constraint of continuity, three vorticity transport equations that describe flow kinetics, and the transport equation for temperature. The problem is well posed even though only $\mathbf{u}$ is specified at the solid wall. However, to successfully solve the momentum equations, a boundary condition for the vorticity vector on the no-slip wall must be specified. The boundary condition dervied from equation (6) to constrain the component of the equation $\mathbf{\Omega} = \nabla \times \mathbf{u}$ normal to S. Hence, the vorticity at a no-slip wall is represented by

$$\mathbf{\Omega}_{wall} = \hat{\mathbf{n}} \cdot \nabla_S \times \mathbf{u} \tag{11}$$

4.2 Problem Translation in PICMSS

The approach to numerics theory expression generalization employs a discrete implementation of a weak form for INS conservation law systems. The weak form is a collection of theory-rich concepts and techniques for the construction of a near-optimal approximate solution processes for the initial-boundary value problem statements presented in computational continuum mechanics.

A truly valuable instruction distinction accrues to use of calculus in weak form implementation. Specifically, in finite element form, all resultant algorithm components are analytically formed (no Taylor series) and are universally expressed as a matrix statement, with all formation processes conducted on a master element domain. Such weak form constructions are captured and formulated to describe the model problems by choosing the appropriate operators and operands defined in the PDE system. Examples of such operators are

$$\nabla^2(?) \ , \ \nabla \times (?) \ , \ \nabla \cdot (?) \ , \ \frac{\partial(?)}{\partial t} \ , \ \{(?) \cdot \nabla\}(?) \tag{12}$$

where (?) can be any operand or defined variables of the PDE system. Only one operand (right-hand side) is needed for linear operators. An additional left-hand side operand is required to implement the nonlinear operators.

The finite element weak form approximation of the state variable is,

$$\mathbf{q}(\mathbf{x}, t) \approx \mathbf{q}^h \equiv \sum_{j=1}^{m} N_j(\mathbf{x})^t \mathbf{Q}_j(t), \tag{13}$$

where $\mathbf{q}$ represent state variables $\{\mathbf{u}, \mathbf{\Omega}, \mathbf{\Theta}, \mathbf{P}\}$ at nodes of basis functions $\{N\}$. The operators are then represented by element-rank matrices of the following forms,

$$\nabla^2 = \int_{\sigma_e} \nabla\{N\} \cdot \nabla\{N\}^T dV, \tag{14}$$

$$\frac{\partial}{\partial x} = \int_{\sigma_e} \{N\} \frac{\partial \{N\}^T}{\partial x} dV, \tag{15}$$

$$\frac{\partial}{\partial y} = \int_{\sigma_e} \{N\} \frac{\partial \{N\}^T}{\partial y} dV, \tag{16}$$

$$\frac{\partial}{\partial z} = \int_{\sigma_e} \{N\} \frac{\partial \{N\}^T}{\partial z} dV, \tag{17}$$

$$\frac{\partial \{Q\}}{\partial t} = \int_{\sigma_e} \{N\}\{N\}^T dV \{Q\}, \tag{18}$$

The nonlinear terms, $\{(?) \cdot \nabla\}(?)$, will consist of a combination of the following integrals,

$$\{?\}\frac{\partial}{\partial x} = \{?\} \int_{\sigma_e} \{N\} \frac{\partial \{N\}}{\partial x} \{N\}^T dV, \tag{19}$$

$$\{?\}\frac{\partial}{\partial y} = \{?\} \int_{\sigma_e} \{N\} \frac{\partial \{N\}}{\partial y} \{N\}^T dV, \tag{20}$$

$$\{?\}\frac{\partial}{\partial z} = \{?\} \int_{\sigma_e} \{N\} \frac{\partial \{N\}}{\partial z} \{N\}^T dV. \tag{21}$$

PICMSS adapts this methodology to prescribe the PDE which models the physics of a problem supplied by the user.

4.3 Results for Channel Flows

Fully developed, and developing flow in a straight rectangular channel tests validity and accuracy of the algorithm boundary conditions and the constraint of continuity. The verification steady state fully-developed axial (u)-velocity distribution is [5]

$$u = \frac{48}{\pi^3} \frac{\xi_u(y,z,h)}{\varphi} \tag{22}$$

$$\xi_u(y,z,h) = \sum_{n=1,3,5}^{N} (-1)^{\frac{n-1}{2}} \left[1 - \frac{\cosh(n\pi y/2h)}{\cosh(n\pi/2)}\right] \frac{\cos(n\pi z/2h)}{n^3} \tag{23}$$

$$\varphi = 1 - \frac{192}{\pi^5} \sum_{n=1,3,5}^{N} \frac{\tanh(n\pi/2)}{n^5} \tag{24}$$

where h is the duct half-height and N is a large integer, e.g, $N = 200$ is used.

Solutions of fully developed straight channel flows in 2D and 3D were established for timing and verified with exact solutions. The GMRES Krylov sparse iterative solver, in conjunction with the least square preconditioner, was selected to solve the matrix statement. Computation were carried out on the Compaq supercluster at the Oak Ridge National Laboratory and a PC Cluster with Gigabit Ethernet interconnect at the University of Tennessee, Knoxville. Results of timing analyses for the 2D case of mesh size M=45x89 and the 3D case of mesh size M=15x15x51 using the pressure projection INS formulation are shown in Tables 1 and Table 2. Timing for the 3D case of mesh size M=24x24x25 using the velocity-vorticity formualtion is shown in Tables 3.

Superlinear speedup is observed in Table 3. Such phenomenon is not unusual for the treatment of large sparse matrices. It is primarily due to the result of more efficient utilization of cache on local processors. The selected problem size is too large to be solved using three processors.

Table 1. : Timing Analyses, Steady 2D Straight Channel Flow, M=45x89, Pressure Projection INS Formulation

Machine Platform	Number of Processors	Last Time Step Assemble(sec)	Last Solve(sec)	Total time(sec)	Speed Up
Compaq	3	2.374	1.463	244. 2	1.00
Compaq	5	0.957	1.058	138.9	1.758
Compaq	15	0.190	.992	80.2	3.044
PC	3	5.73	8.1	858.0	1.00
PC	5	2.428	5.91	517.0	1.66
PC	15	0.510	2.73	220.0	3.9

Table 2. : Timing Analyses, Steady 3D Straight Channel Flow, M=15x15x51, Pressure Projection INS Formulation

Machine Platform	Number of Processors	Last Time Step Assemble(sec)	Last Solve(sec)	Total time(sec)
Compaq	15	9.8726	1.555	754.3
PC	15	24.96	8.14	1980.5

5 Conclusions and Future Work

An interoperable parallel computational platform, PICMSS, has been designed to facilitate rapid simulation of engineering systems involving non-linear coupled fluid/thermal processes. It serves as a collocation point for incorporating novel computational methodologies as well as state-of-the-art advances in communication and information sciences. Expansion to assimilate new and innovative algorithm constructions is intrinsic to the design of PICMSS. Installation

Table 3. : Timing Analyses, Steady 3D Straight Channel Flow, M=24x24x24, Vorticity-Velocity Formulation

Machine Platform	Number of Processors	Last Time Step Assemble(sec)	Last Solve(sec)	Total time(sec)	Speed Up
Compaq	3	1034.4	28.1	11722.2	1.0
Compaq	6	234.6	15.58	2776.9	4.2
Compaq	12	78.63	5.77	944.5	12.4
Compaq	24	31.27	3.00	395.3	29.6

of PICMSS on the Internet metacomputing environment will be accomplished via the NetSolve system, which will be enhanced and adapted to manage and monitor computational resources for the proposed PICMSS environment.

The future goal is to advance and proliferate the Parallel Interoperable Computational Mechanics Simulation System capable of solving multi-disciplinary problems in engineering design. Upon establishing the first release of PICMSS, an Alliance effort is proposed to support this Internet-based computational environment. Users will be encouraged to expand the physical and numerical capacities of PICMSS as a national software resource.

6 Acknowledgments

The authors would like to acknowledge the support of computer resources from the Computer Science and Mathematics Division and the Solid State Division at Oak Ridge National Laboratory.

References

[1] Balay, S., Gropp, W. D., McInnes, L. C., and Smith, B. F., The Portable, Extensible Toolkit for Scientific Computing, Version 2.0.24, Argonne National Laboratory, 1999.
[2] Casanova, H., Dongarra, J. and Seymour, K., Users' Guide to NetSolve, Department of Computer Science, University of Tennessee, 1998.
[3] Foster, I. and Kesselman, C. eds, The Grid: Blueprint for a New Computing Infrastructure, San Francisco, Morgan Kaufman Publishers, 1999.
[4] Hutchinson, S., Prevost, L., Shadid. J., and Tuminaro, R., Aztec Users' Guide, Version 2.0 Beta, Massively Parallel Computing Research Laboratory, Sandia National Laboratory, 1998.
[5] White, T., Viscous Fluid Flow, New York, McGraw-Hill, 1974.
[6] Wong, K. L., A Parallel Finite Element Algorithm for 3D Incompressible Flow in Velocity-Vorticity Form, PhD Dissertation, The University of Tennessee, 1995.