This is the published version (version of record) of:


Available from Deakin Research Online:

http://hdl.handle.net/10536/DRO/DU:30004864

Copyright: ©2002 IEEE. Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works must be obtained from the IEEE.
The Design and Development of An Integrated System for Object-Oriented Finite Element Computing

Yang Xiang and Wanlei Zhou
School of Computing and Mathematics, Deakin University, Burwood 3125, Victoria, Australia
{yang, wanlei}@deakin.edu.au

Abstract

Finite Element Method (FEM) is widely used in Science and Engineering since 1960's. The vast majority of FEM software is procedure-oriented. However, this conventional style of designing FEM software encounters problems in maintenance, reuse, and expansion of the software. Recently the object-oriented finite element method attracts the attention of lots of researchers, and now there is a growing interest in this method. In this paper, the object-oriented finite element (OOFE) is briefly introduced. Then the design and development of an integrated OOFE system is described. A comparison of the integrated OOFE system and a procedure-oriented system shows that our OOFE system has many advantages.

1. Introduction

1.1 Overview of Finite Element Method

Computing technology nowadays makes it possible to solve complicated problems that are difficult to work out. Many problems that could hardly be figured out before can now be solved by numerical methods. One such a significant method is the Finite Element Method (FEM).[4]

FEM was developed in the 1960’s when the computing facilities were very limited and the programming languages were still juvenile. Almost every Finite Element (FE) program developed at the time was procedure-oriented and written in FORTRAN.

1.2 Problems in Traditional FEM

A procedure-oriented finite element program may cost years to develop; yet the reusability of the program is not satisfiable. When a new algorithm or a new kind of element is designed it is difficult to apply it in the software because the alternation of one subroutine may affect the whole program. The risk of errors being generated may be increased greatly. So it is hard to maintain and reuse the traditional FE software because the software engineer has to be in control of every particular detail of the whole program.

Moreover, the pre-processing is always the bottleneck of applications of the FE technology. An effort is being made by developers to supply analysis programs as black box modules to be used with CAD systems. However, conventional programs are not easily reshaped to perform the desired tasks, so developers must redesign and rebuild their finite element libraries. When the FE engineering software system becomes more and more complex, the FE software crisis appears.

1.3 Merits of the Object-Oriented Finite Element Method

Fortunately, coupled with the development of hardware has been the development of software and
programming languages. With lots of merits, such as data abstraction and encapsulation, inheritance and polymorphism, the object-oriented technology reflects the concerns of software developers. Then the object-oriented approach becomes attractive for the FE engineering software development. In the last few years many articles on Object-Oriented Finite Element (OOFE) research have been published. Forde, et al., published one of the first publications of the OOFE in 1990[1]. Then other authors increased their awareness of the advantages of object-oriented method in finite element software development. [2][3][5][6][8]

1.4 Objectives of Research

Currently, the work in object-oriented finite element method has been largely concentrated on identifying the FE object abstraction, application in some special areas, and so on. However, the existing research is weak in the following two areas.

- Firstly, the work to date has largely ignored discussing how to integrate the pre-processor, solver and post-processor into a practical software system. Present research mainly focuses on how to build a finite element analysis system by using object-oriented method without the pre-processor and post-processor. However, a concise and fully functional FE system is needed because it has lots of advantages.

- Secondly, few researchers have concerned about the computing efficiency of OOFE. Although the merits of object-oriented programming attract researchers, to some extents, by using object-oriented method the computing efficiency of an FE program may lose.

This paper addresses the two areas mentioned previously. The objectives of this research are to develop an OOFE system that integrates the components that FE software requires.

2. Object-Oriented Analysis

The object-oriented finite element (OOFE) method is to use object-oriented method to build up FE software systems. OOFE software needs to transfer the real world situation into an object model by the means of abstraction, analysing the object, establishing the layer structures of the classes and developing the system frame.

We have developed an OOFE system called MEG+. It consists of three main parts: the pre-processor, the finite element solver and the post-processor. The pre-processor is a kind of CAD system, which is used to build the model. It is actually not only a CAD system, but also a mesh generator that has the self-adaptation ability. The mesh it generated includes the structured mesh and the unstructured one. This part of MEG+, the pre-processor, called MEG, can be download from website http://www.meg-tech.com.

The software architecture of MEG+ has the object-oriented style based on data abstraction and object-oriented organization. Data representations and their associated primitive operations are encapsulated in an abstract data type.

The objects in OOFE can be classified into four categories as follows, Mathematical classes, FE-Object classes, CAD classes and Control classes. The following sections outline the components of some of these classes.

2.1 Mathematical classes

The mathematical classes include Matrix, Vector, EquationSolver, and so on. These classes are used to build up and solve the finite element equations.

2.2 FE-Object classes

The FE-Object classes include Domain, Mesh, BackMesh, Element, Node, GaussPoint, DOF (Degree Of Freedom), Constraint, Material, Load, and so on. The FE-Object classes are the most important and comprehensive classes.
• **Domain.** A geometric concept, the Class Domain, represents the field of FE analysis. It is built on multiple inheritances and consists of sub-classes such as SubDomain. SubDomain is divided into FE mesh, which includes the elements and the nodes.

• **Mesh and BackMesh.** The Class Mesh is the core class in the pre-processor. It generates the mesh data automatically, stores the data to the database, and draws the mesh in the CAD system. In order to control the mesh quality, the BackMesh class is employed. BackMesh will be hidden from the user. After the FE mesh is generated and checked in the graphical interface, the data of mesh, the element and node data will be transferred to the database.

• **Element.** As an abstract basic class, the Element consists of different element types, for instance, the brick element, shell element, beam element, etc. In OOFE, the work of coding is greatly decreased in element design because of the inheritance of object-oriented technology. When the basic element class is set up, other subclass elements can be easily added. The attributes of class Element include the element number, type, node information, etc and the functions are calculation of stiff matrix, calculation of stress or strain matrix, calculation of Jacobi matrix, etc. It also can store the data to the database.

• **Node.** The Node class has some functions of input and output, and other calculations. Moreover, the number of nodes, DOF information of the node, coordinate system of nodes are the attributes of this class.

• **Material.** The attributes of the Material class include the number of materials, the material parameters, etc. The functions include input and output the data, modify the data, etc. The Material class also has the link between the graphical interface and the database.

• **Load.** The FE analysis includes several load steps and different load types. The Load class is similar to the Element class because it has the basic class and the particular load classes. In the upper level, the Load class can be classified into NodeLoad and ElementLoad. NodeLoad is the load applied on the node, for example, the convergence force on a node, or temperature on a node. ElementLoad includes surface load, body load, etc. The control classes control the load because in the whole analysis the load may be divided into lots of steps and applied at a different time.

### 2.3 CAD classes

The CAD classes include Point, Line, Polygon, Text, etc. In a 3D situation, these classes may include other solid objects. In MEG+, all the input of an FE user can be done in the CAD system.

### 2.4 Control classes

The Control classes include GlobalControl, Time, and so on. The Control classes are used to control the non-linear and time stepping.

• **GlobalControl.** The GlobalControl class plays the key role of the FE analysis. It controls the analysis type, for example, the linear analysis or non-linear analysis and controls the global time stepping. So generally speaking, it controls the FE analysis.

• **Time.** The Time class is applied in time stepping. The attributes of Time class are the length of the time stepping, the number of stepping, etc. It has links with the database.

The MEG+ system includes three parts, the FE-Object classes, the CAD classes and the Control classes. The Control classes are the centres to control the process and exchange information between different parts. This part controls the mesh auto-generation and the solver. The FE-Object classes can manage the data by themselves.
3. OOFE system implementation and example

FE software includes three parts of functions: pre-processor, solver and post-processor. In this section, these three parts are discussed to demonstrate the application of OOFE and show its advantages. In MEG+, we use a compressed stiffness matrix storage scheme. By using this scheme, the computing performance is improved greatly.

Figure 1. Mesh generated by MEG+

Figure 2. Stress contour by MEG+

A 2D plane strain problem is solved by MEG+, where a dam is applied with water pressure and gravity. The purpose is to compute the displacement and stress condition. The analysis result shows the stress in the dam body. The mesh generated by pre-processor of MEG+ is shown in figure 1. The 3rd principal stress contour computed by MEG+ is shown in figure 2.

4. Conclusions

An integrated object-oriented FE software named MEG+ is described in this paper. The design and implementation of the object-oriented FE software including a pre-processor, the FE solver, and a post-processor are presented.

Object-oriented FE software has lots of advantages in maintenance, reusability, code readability, expendability, fast prototyping, computing efficiency, error tolerance, etc. In this paper we have shown that, the FE engineering software integrating the entire pre-processor, FE solver, and the post-processor, and containing reusable components or graphical interface is a better solution than the traditional FEM.

References