第 21 卷 第 3 期 岩石力学与工程学报 21(3)：404～409
2002 年 3 月 *Chinese Journal of Rock Mechanics and Engineering* *March*，2002

岩土工程中的面向对象有限元程序设计

项 阳 平 扬 葛修润
(中国科学院武汉岩土力学研究所岩土力学重点实验室 武汉 430071)

摘要 介绍了面向对象的程序设计方法，按照这种方法，建立了有限元模型的类，给出了类的 C^{++}语言描述和它的实现方法，并设计了基于 Windows 95 / 98 / NT 平台的岩土工程有限元软件。相关的类和方法包括处理矩阵的类、单元类、节点类、高斯点类、形函数类、应力类、集中力类、分布力类、绘图点类、绘图线类、网格类、区域类、边界类、文字类等等。给出了基本的程序框架和有关的程序段。

关键词 面向对象，有限元，程序设计，C^{++}
分类号 TU 75，O 242.21，TP 311 **文献标识码** A **文章编号** 1000-6915(2002)03-0404-06

1 引 言

传统的有限元程序大多用面向过程(Procedure Oriented)的编程语言编写，例如 FORTRAN。这类编程语言着重考虑的是对数学公式的翻译。在过去的几十年里，面向过程的编程方法得到了充分的发展和应用，逐渐形成了一套结构化问题分析、系统设计和程序编制等一系列的软件开发技术。然而，随着软件规模的扩大，面向过程的编程语言由于不能提供数据抽象、数据封装等概念，开发和维护软件中存在的潜在困难显得日益突出，如代码重新利用率较低、调试较复杂、开发周期长等等。面向对象(Object Oriented)的编程方法在近年来发展迅速，已逐渐成为编程领域的主流。本文利用面向对象的程序开发语言 C^{++}，根据有限元的基本原理，建立了有限元模型的类，并设计了基于 Windows 95/98 平台的岩土工程有限元软件。结果表明，用这种方法来设计有限元软件，设计和调试周期较传统方法明显缩短，代码的重新利用率也明显提高[1, 2]。本文研究了面向对象有限元程序设计方法，并设计了基于 Windows 95 / 98 / NT 平台的岩土工程有限元软件，并用算例进行了验证。

2 面向对象编程的基本概念

2.1 对象、类和消息(object，class and message)

对象把客观世界的实体和计算机系统运行实体有机地结合在一起。从存储的角度来看，对象拥有一块存储区，其中既有数据，又有方法。相似的对象归并到一个类中，它将这些对象的共同特征集中起来，从而使系统在结构上形成了一个具有特定功能的模块和一种代码共享的手段。消息也是面向对象的一个基本机制。在面向对象的程序设计系统中，通过消息来请求对象的动作，对象间的相互联系也通过消息来完成。

2.2 数据抽象(data abstraction)

数据抽象是指从较特殊的类或对象中抽出一般属性以建立一个超类(Super Class)的过程。 在这个过程中，研究目标程序要解决的问题和组成该问题的概念性的实体，并在不同的层次上进行抽象。

2.3 封装(encapsulation)

封装又称数据隐藏，就是将方法和数据放在同一对象中，使得对数据的存取只能通过该对象本身的方法来实现，程序的其他部分不能直接作用于此数据。这一特性大大地降低了模块间的耦合性，从而提高了程序的可靠性，尽可能地排除了对数据进行任意访问造成的隐患。

2.4 继承性(inheritance)

继承性就是指一个类可以继承其父类的所有数据和成员函数，同时又可以定义自己的数据和成员函数。这使得编程工作大为减轻。因为可以使用

2000 年 2 月 29 日收到初稿，2000 年 4 月 25 日收到修改稿。
作者 项 阳 简介：男，1975 年生，2000 年于中国科学院武汉岩土力学研究所岩土工程专业获硕士学位，现为澳大利亚 Deakin 大学访问学者，主要从事工程软件方面的研究工作。

某类来完成一些普通的工作，而用特定的类来完成特定的工作。

2.5 多态性(Polymorphism)

多态性是指同一消息被不同的对象接收后解释为不同含义的能力。由于这种特性，不同的对象接收到用户统一发送的消息就可完成不同的工作。

3 面向对象程序设计

3.1 岩土工程的面向对象有限元程序分析

岩土工程中的面向对象有限元程序既有一般面向对象程序的特点，也有其自身的特点。它需要考虑开挖、衬砌、回填等等施工步骤，也要设计节理单元、锚杆单元等等特殊单元。一般有限元分析的过程是从单元分析开始的。有限元计算的目的就是要把整个区域离散化，把计算转化到单元上来完成[3]。所以面向对象有限元程序的类的设计[4]可以有：(1) 总体有限元类，它的数据有单元数、节点数、材料数、施工步数等等。(2) 单元类，它的数据有材料特性、节点信息、荷载条件、约束条件、高斯点类、形函数类、应力类等等。在岩土工程中，通常可使用 3，4，8 节点等参单元。另外，还有一些特殊的单元，如节理单元，锚杆单元等。(3) 节点类，它的数据有节点坐标、节点自由度、节点力、约束条件、节点位移等等。(4) 矩阵类，专门用来处理各种矩阵，如刚度矩阵、应力-应变矩阵、位移矩阵等等。(5) 网格类，用来生成有限元网格。(6) 绘图类，在前处理和后处理中都要用到。

下面给出包含注释的 C^{++}描写的程序段。

3.2 总体类

有限元总体类的定义如下：

```
class COOFEM：public CObject
{
public：//构造函数和析构函数
    DECLARE_SERIAL (COOFEM)
    COOFEM();
    COOFEM(int iNumElement，int iNumNode，
        int iType，int iNumMaterial，int iStep);
    virtual～COOFEM();
public：//串行化和初始化
    void Serialize(CArchive & ar);
    void Initialize(int iNumElement，int iNumNode，
        int iType，int iNumMaterial，int iStep);
private：
    int iNumElement；          //单元数
    int iNumNode；             //节点数
    int iType；                //问题类型
    int iNumMaterial；         //材料数
    int iStep；                //施工步数
private：
CMesh* pMesh；          //指向网格类的指针
CElement* pElement；    //指向单元类的指针
CNode* pNode；          //指向节点类的指针
CMaterial* pMaterial；  //指向材料类的指针
private：
CMatrix GlobalStiffMatrix；  //总体刚度矩阵
CMatrix GlabalLoadMatrix；   //总体载荷矩阵
CMatrix GlobalDispMatrix；   //总体位移矩阵
public：
int OutputAll()；           //输出所有结果数据
};
```

总体类的初始化传入基本数据单元数、节点数、材料数、施工步数，在构造函数中为网格类、单元类、节点类、材料类分配内存空间。OutputAll()函数输出所有的结果数据。

3.3 单元类

单元的定义如下：

```
class CElement：public Cobject
{
public：//构造函数和析构函数
    DECLARE_SERIAL (CElement)
    CElement();
    Virtual～CElement();
public：//串行化和初始化
    void Serialize(CArchive & ar);
    void Initialize();
private：//单元的特性数据和单元数
    ElementData e[MAX_ELEM_NUM];
    int iNumElement;
public：
    CNode* GetNode();       //获取指向节点的指针
    int GeneStiff(Matrix* pGlobalStiffMatrix);
    int CalStress(ElementData e);
    int CalStrain(ElementData e);
    int CalGauss(ElementData e);
    BOOL bIsExcavated()；//检查是否被开挖
};
```

其中 GeneStiff()为计算刚度矩阵的函数，CalStress()为计算应力的函数，CalStrain()为计算应

变的函数，CalGauss()为计算高斯点特性的函数。ElementData 是一个结构体，它包括单元的基本数据。它的定义如下：

```
typedef struct
{
    int Points[NUM_POINTS];    //单元节点数
    int Material;              //材料号
    int DamagedType;           //破坏类型
    int iExcavate;             //开挖步骤
    float SigmaX[MAX_STEP];// X 方向的主应力
    float SigmaY[MAX_STEP];// Y 方向的主应力
    float SigmaZ[MAX_STEP];// Z 方向的主应力
    float Sigma1[MAX_STEP];//最大主应力
    float Sigma2[MAX_STEP];//最小主应力
    float Tao[MAX_STEP];    // 剪应力
    float Angle[MAX_STEP];  //应力方向
    int Type;               //单元类型
} ElementData;
```

其他类型的单元诸如锚杆单元、节理单元都可以从这个基本单元类派生得到。另外，当单元为开挖单元时，iExcavate 变量设置为开挖步数。

3.4 节点类

节点类的定义如下：

```
class CNode：public CObject
{
public:    //构造函数和析构函数
    DECLARE_SERIAL( CNode )
    CNode();
    virtual ～CNode();
public:    //串行化和初始化
    void Serialize(CArchive & ar);
    void Initialize();
public:
    float GetMaxX();    //获得 X 方向的最大坐标
    float GetMinX();    //获得 X 方向的最大坐标
    int GetBC(int iIndex);    //获得节点的约束状态
    int SetBC(int iIndex);    //设置节点的约束状态
    void OutputAll();         //输出所有信息
    void OutputDisp();        //输出位移
    void OutputForce();       //输出节点力
private:    //节点的特性数据和节点数
    NodeData n[MAX_NODE_NUM];
    int iNumNode;
};
```

NodeData 是一个结构体，它包括节点的基本数据。它的定义如下：

```
typedef struct
{
    int iIndex;                    //序号
    float X;                       // X 方向坐标
    float Y;                       // Y 方向坐标
    float UX[MAX_STEP];            // X 方向位移
    float UY[MAX_STEP];            // Y 方向位移
    CNodeForce Force[MAX_STEP];    //节点力
    int iFixed;                    //约束信息
    int iExcavated;                //开挖信息
}NodeData;
```

节点类的函数可以处理它的基本数据。

3.5 矩阵类

在有限元计算中，矩阵的运算是非常重要的。而且它的刚度矩阵存在大量非零元素，与一般矩阵的处理方法也有不同。为了节省内存，总体刚度矩阵按照一维存储，只存放非零元素和它的位置索引。

表示矩阵的类为：

```
class CMatrix
{
public:    //构造函数和析构函数
    CMatrix();
    CMatrix(int iNumRow，int iNumColumn);
    CMatrix(const CMatrix&);
    ～CMatrix();
protected:
    int iNumRow;            //矩阵的行数
    int iNumColumn;         //矩阵的列数
    MatrixData* pCurrent;   //指向当前矩阵元素的指针
    float& Insert(MatrixData* pElem，int iRow，
            int iColumn);   //插入一个元素
public:    //获得当前元素的行号和列号
    int GetRow(MatrixData* pCurrent);
    int GetColumn(MatrixData* pCurrent);
public:    //重载运算符
    virtual float& operator(int iRow，  int iColumn);
    virtual CMatrix& operator +=(CMatrix&);
    virtual CMatrix& operator =(CMatrix&);
    virtual CMatrix& operator *=(CMatrix&);
    virtual CMatrix operator *(float，CMatrix&);
    virtual CMatrix operator *(CMatrix&，float);
    virtual CMatrix operator *(CMatrix& ，
CMatrix&);
```

```
    virtual CMatrix operator +(CMatrix& , CMatrix&);
}
```

MatrixData 是一个结构体，它包括矩阵的基本数据。它的定义如下：

```
typedef struct
{
    int iRow;                   //行号
    int iColumn;                //列号
    float Value;                //元素值
    MatrixData* pNext;  //指向下一个元素的指针
    MatrixData* pPre;   //指向上一个元素的指针
    MatrixData* pHead;  //指向第一个元素的指针
}MatrixData;
```

矩阵类重载了运算符()，+=，=，*，+，对矩阵的操作可以和对一般数一样方便地操作。如：

```
MatrixData A，B，C;
C=A*B;
A+=B;
```

3.6 网格类

网格类是有限元前处理非常重要的类，本文设计的网格类具有自动生成网格、存储网格数据、绘制网格等等功能。

```
class CMesh：public CObject
{
public：  //串行化和初始化
    DECLARE_SERIAL( CMesh )
    void Serialize(CArchive & ar);
    void Initialize();
public：  //构造函数和析构函数
    CMesh();
    Virtual～CMesh();
private：  //成员数据：节点和单元
    CNode Node;
    CElement Elem;
public：
    void PrepareData();  //为网格生成准备数据
    void Generate(int iMeshType);  //生成网格
    void Modify(BOOL bDense);  //修改网格
    void ControledModify();  //控制修改网格
    BOOL IsJointNode(int i);  //判断是否为节理
                              //单元的节点
    void DrawElemNumber(float startx , float starty，
            int ScreenHigh，float blc，CDC* pDC);
    void DrawNodeNumber(float startx , float starty，
            int ScreenHigh，float blc，CDC* pDC);
    void DrawDeformation(int iStep，float startx，
            float starty，int ScreenHigh，
            float blc，CDC* pDC);
    void DrawBC(float x，float y，float startx，
            float starty，int ScreenHigh，
            float blc，CDC* pDC);
    void DrawElement(int iIndex，float startx，
            float starty，int ScreenHigh，
            float blc，CDC* pDC);
    void DrawResult(float startx，float starty，
            int ScreenHigh，float blc，CDC* pDC);
    void DrawAll(float startx，float starty，
            int ScreenHigh，float blc，CDC* pDc);
                //以上函数为绘图的函数
};
```

3.7 其　他

作为用 Visual C^{++}开发的程序，视类和文档类是必不可少的类。MFC 的视窗文档结构使软件的设计更为方便，使得开发者可以集中精力研究面向对象有限元的部分。

关于大规模有限元方程组的解法，由于岩土工程的特殊性，方程组通常都有病态性。为此，本文采用了共扼梯度法来求解方程。设计了求解器类 CSolver。求解方程组的函数如下：

```
float* CSolver ：：ComputeSolution(float* b，
                        int n，float* rsq)
{
    int j，iter，irst=0;
    float aden，anum，bsq，dgg，eps2，gam，gg，rp;
    float* g，*h，*xi，*xj，*vector()，*x;
    g=vector(1，n);
    h=vector(1，n);
    xi=vector(1，n);
    xj=vector(1，n);
    eps2=n*EPS*EPS;
    while(TRUE)
    {
        ++irst;
        asub(x，xi，n);
        rp=bsq=0.0f;
        for(j=1；j<=n；j++)
```

```
{
        bsq+=b[j]*b[j];
        xi-=b[j];
        rp+=xi[j]*xi[j];
}
atsub(xi，g，n);
for(j=1；j<=n；j++)
        h[j]=g[j]=-g[j];
for(iter=1；iter<=10*n；iter++)
{
        asub(b，xi，n);
        anum=aden=0.0;
        for(j=1；j<=n；j++)
        {
                anum+=g[j]*h[j];
                aden+=xi[j]*xi[j];
        }
        if(aden==0.0)    JumpOut(0);
        anum/=aden;
        for(j=1；j<=n；j++)
        {
                xi[j]=x[j];
                x[j]+=anum*h[j];
        }
        asub(x，xj，n);
        *rsq=0.0f;
        for(j=1；j<=n；j++)
        {
                xj[j]-=b[j];
                *rsq+=xj[j]*xj[j];
        }
        if(*rsq==rp||*rsq<=bsq*eps2)
        {
                FreeAll();
                return x;
        }
        if(*rsq>rp)
        {
                for(j=1；j<=n；j++)
                     x[j]=xi[j];
                if(irst>=3)
                {
                     FreeAll();
                     return x;
                }
                     break;
                }
                rp=*rsq;
                atsub(xj，xi，n);
                gg=dgg=0.0f;
                for(j=1；j<=n；j++)
                {
                     gg+=g[j]*g[j];
                     dgg+=(xi[j]+g[j])*xi[j];
                }
                if(gg==0.0)
                {
                     FreeAll();
                     return x;
                }
                gam=dgg/gg;
                for(j=1；j<=n；j++)
                {
                     g[j]=-xi[j];
                     h[j]=g[j]+gam*h[j];
                }
           }
           JumpOut(1);
      }
}
```

计算问题的描述就变得非常直观，只需要如下语句就可以实现：

```
CSolver CurrentSolver;
FloatArray* x;
x=CurrentSolver.CumputeSolution(b，n，rsq);
```

4　算　例

某深基坑开挖工程的整体稳定性分析的网格定义及变形如图 1 所示。图中所示的变形结果是安全系数为 1.30 时的计算所得，按平面应变问题分析。力学参数 E =13 MPa，μ = 0.31，c =1 MPa，φ = 26°。本文提供的面向对象有限元程序是正确可靠的，计算结果和 Zace Services Ltd 公司开发的 Z_Soil 4.24 商用软件所计算的结果基本上是一致的。

5　结　论

有限元软件的规模越来越大，所要实现的功能

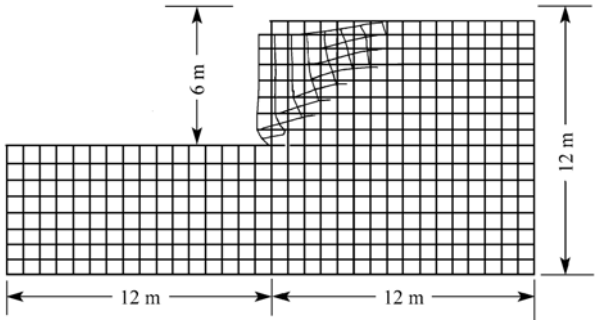

图 1　算例

Fig.1　Example

也越来越多，用面向对象的方法来研究有限元，使得构造一个大规模复杂的有限元应用软件的难度大为降低，开发周期大为缩短。面向对象的方法引入的类，使得编程中的错误限制于局部；在类中加入新的数据和方法，可以不用改动原来的程序；从已构造好的类中继承新类，使代码的利用率大为提高。这也正是面向对象有限元方法的优势所在。通过上面的例子可以看出，面向对象的有限元方法将是颇具潜力的有限元方法发展方向之一。计算的结果表明，本软件的效果是令人满意的。

参 考 文 献

1　崔俊芝，梁　俊. 现代有限元软件方法[M]. 北京：国防工业出版社，1995

2　Machie R I. Object-oriented methods-finite element programming and engineering software design[J]. Computing in Civil and Building Engineering，1995，(1)：133～138

3　Fenves G L. Object-oriented programming for engineering software development[A]. In：Engineering with Computers[C]. New York：Springer Verlag，1990

4　Owen D R J，Hinto E. Finite Elements in Plasticity Theory and Practice[M]. London：Pinerige Press，1980

OBJECT-ORIENTED FEM PROGRAMMING FOR GEOTECHNICAL ENGINEERING

Xiang Yang，Ping Yang，Ge Xiurun

(*Key Laboratory of Rock and Soil Mechanics，Institute of Rock and Soil Mechanics，The Chinese Academy of Sciences，Wuhan*　430071　*China*)

Abstract　The object-oriented programming method is introduced. By this method，the classes of finite element models are designed and their C^{++} description and the implementation are given. A geotechnical engineering FEM software based on Window95/98/NT is designed and developed. The common FEM classes are：the class of matrix，element class，node class，Gauss class，shape function class，stress class，load class，distributed force class，the drawing class，mesh class，region class，boundary class，word class，etc. The basic program frame and the codes are given.

Key words　objec-oriented，finite element method，programming，C^{++}

中国科学院武汉岩土力学研究所岩土力学实验室被中国科学院批准为院重点实验室

2002 年 1 月 4 日，中国科学院武汉岩土力学研究所岩土力学实验室被中国科学院正式批准为重点实验室。岩土力学重点实验室是中国科学院开展岩土力学应用基础研究的综合性研究机构，经过多年的积累和发展，形成了计算与智能岩石力学、环境岩石力学、特殊土土力学 3 个主要学科方向。目前，重点实验室不仅有一支优秀的科研队伍，而且拥有功能齐全的岩土力学试验设备。热诚欢迎国内外的专家、学者来实验室指导工作、访问、讲学和客座研究。

(焦玉勇供稿)