I－11 分散オブジェクト技術によるWebベースの協調的有限要素解析システム

A Web-based Collaborative Finite Element Analysis System Using a Distributed Object Technology

矢吹 信喜[1] 岩崎 充乗[2]
Nobuyoshi Yabuki Mitsunori Iwasaki

【抄録】本研究では，大規模・複雑化している有限要素解析コードの開発・メンテナンスの効率化，高度な要素やソルバの協調的な開発環境の構築，一台のコンピュータに負荷を集中させず分散させること，およびユーザがWWW上で高度なFEM解析を安価に出来るようにすることを目的に，分散オブジェクト技術を利用した有限要素解析システムWebFEMを，オブジェクト指向言語Javaと分散オブジェクト環境HORBにより開発した．本システムでは，複数の各種要素およびソルバをそれぞれサーバに配置し，分散サービスプロバイダを形成し，使用頻度の高いソルバのサーバは複数用意し負荷分散を図っている．

【キーワード】分散オブジェクト技術，分散処理，オブジェクト指向，有限要素解析，WWW

【Abstract】In order to improve the efficiency of development and maintenance of large and complicated finite element analysis codes, to establish a collaborative development environment of advanced elements and solvers, to distribute computational loads to multiple computers, and for users to perform advanced FEM analysis by a minimum cost on the World Wide Web, we developed a FEM analysis system, WebFEM, by the object-oriented programming language, Java, and distributed computing environment, HORB. In this system, multiple elements and solvers reside in servers, forming a distributed service providers, and multiple servers are provided for a set of frequently used solvers to decrease loads.

【Keywords】Distributed object technology, distributed computing, object-oriented, finite element analysis, WWW

1. はじめに

1980年頃までは有限要素解析（FEM）コードは，各企業や機関における自主開発が基本であったが，その後，最新のアルゴリズムや要素技術を取り入れ，大規模なFEMコードを開発・メンテナンスしていく事がだんだんと困難になっていった．それと同時に米国を中心に我が国でも，商用FEMコードが台頭し，信頼性が向上すると共に，商用コードの購入・使用が一般的になった．

その結果，FEM解析が容易に実施できるようになったが，自らは必要とするが商用コードがサポートしていない要素や機能の開発が難しくなり，一部の商用コードを除けば，必要な時にすぐに対応できないといった問題が発生してきた．さらに，商用コードは，大規模なソフトの開発やメンテナンスを行う事から高価格になり，こうした高価格な商用FEMパッケージを揃えられるのは一部の組織等に限られ，揃えることが出来ない組織とのギャップが顕在化してきたと思われる．

大規模FEMコードの開発・メンテナンスコストを大幅に低減するためには，従来のFortran77に代表されるような手続き型言語からオブジェク

1 正会員 Ph.D. 室蘭工業大学工学部建設システム工学科 助教授 〒050-8585 室蘭市水元町27－1
TEL: 0143－46－5219 FAX: 0143－46－5218 Email: yabuki@news3.ce.muroran-it.ac.jp
2 学会員 室蘭工業大学大学院工学研究科建設システム工学専攻

ト指向プログラミング言語に変更し，コード（オブジェクト）の中身を見なくても容易に再利用できるような環境を構築していく必要があると考えられる．オブジェクト指向プログラミング技術による FEM コードの開発方法に関する研究は，矢川ら[1]や Archer ら[2]等によって既になされているが，オブジェクト指向技術の柔軟性もあって未だ統一的な方法は見当たらないようである．しかし，オブジェクト指向技術によるプログラム開発過程においては，クラスやオブジェクトの振舞いや機能を表記する標準的な方法として，UML[3]（Unified Modeling Language）が広く使用されるようになってきている．

また，FEM コードをサーバやサーバと連動する他のコンピュータなどに分散して配置することにより負荷が１台に集中するのを防ぎ，ユーザはインターネット上のクライアントとしてサーバへアクセスして，適切な課金等の方法により FEM 解析が可能なシステムを構築していくことが望まれる．

さらに，Linux の開発のように多数の開発協力者が最新の要素を取り入れたコード（オブジェクト）を開発し，協調的に FEM コードの高度化を図っていく事が望まれる．この場合，開発したコードを１箇所にまとめて置くのではなく，各開発技術者のコンピュータに分散して配置した方が，個人の知的財産を保護しながら，サーバからのリクエストにより，オンラインで計算を実行することが出来るようになると考えられる．実際，Peng ら[4]は，分散オブジェクト技術を使用して，こうした実行環境に関する研究を行っている．

本研究では，まずトラス要素，フレーム要素および３角形要素を含む２次元静的弾性 FEM コード“ObjFEM”をオブジェクト指向技術により開発した．開発過程ではダイアグラムの表記に UML を，プログラミング言語としては Java を使用した．次に，分散オブジェクト技術により，各要素および連立方程式を解く各種「ソルバ」等のコード（オブジェクト）をそれぞれ異なるコンピュータに分散的に配置し，インターネット上で動作する“WebFEM”を開発した．分散オブジェクト技術としては，ORB（Object Request Broker）の一つである HORB[5] を使用した．さらに，WebFEM を使用して適当な有限要素解析を実施し，本プロトタイプシステムの検証を実施した．

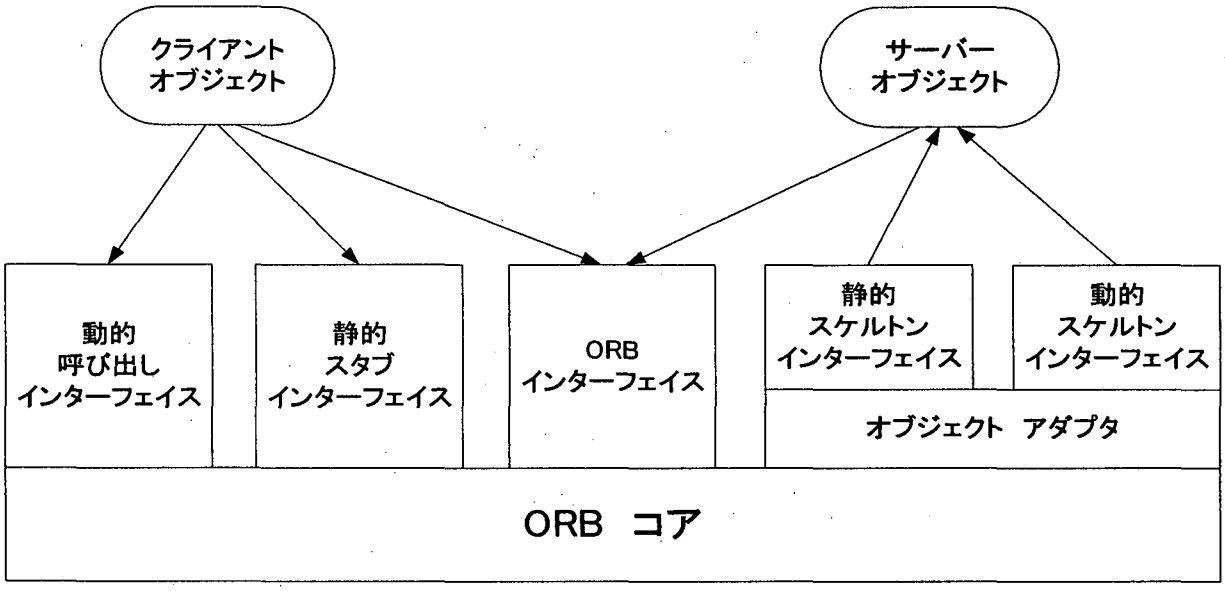
2. 分散オブジェクト技術

オブジェクト指向技術は，当初は人工知能におけるフレームとソフトウェア工学のデータ構造を源にしたプログラミング技術の一つであったが，その後ソフトウェアの分析や開発技術にまで発展してきている．オブジェクト技術の基本は，再利用し易いように相互の関係を系統的に表現し，中身を見えなくしたコード（オブジェクト）にメッセージを送ると動作する，ということだと考えられる．

一方，分散処理技術は，コンピュータを単体としてだけではなく，複数のコンピュータを通信技術によりつなげて利用しようという発想から，クライアント／サーバシステムへと発展し，さらにインターネットの解放に伴い，地理的，組織的な壁が取り払われた地球規模的計算機環境における処理技術と位置付けられると考えられる．

ネットワーク上にある複数のコンピュータにそれぞれ異なったオブジェクトがあるとき，あるマシンからメッセージを別のマシンのオブジェクトに送れば，計算を実行し，答えを返すようにするという，オブジェクト指向技術と分散処理技術が融合した分散オブジェクト技術の発生は自然な流れといえよう．特に，ソフトウェアの開発過程においては，複数のシステムエンジニアが別々にオブジェクトを開発する場合，分散しているオブジェクトを１ヶ所に集結させるより，分散オブジェクト技術により協調的に開発を行う方が効率的だと考えられる．

ネットワーク上の離れた別のコンピュータにあるプログラムを動作させるためには，遠隔手続き呼び出し（RPC：Remote Procedure Call）が必要である．従来，RPC はハード毎に個別であったが，オブジェクト技術を用いて統合化し，異機種

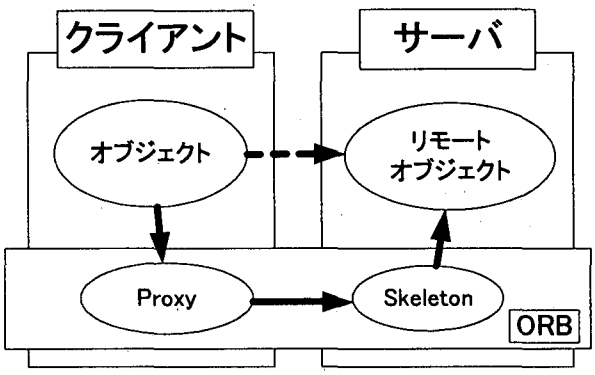


図－1　CORBA の基本アーキテクチャ

コンピュータ間で相互運用出来るようにするために開発されたのが ORB である．さらに，業界団体である OMG（Object Management Group）は ORB を標準化し，CORBA（Common Object Request Broker Architecture）としてその仕様を発表している．CORBA の基本的なアーキテクチャを図－1 に示す[6]．

図－1 のインターフェースの部分は，仕様が統一された IDL（Interface Definition Language）によってプログラムを作成すればよいのだが，相当に難解である．そこで，産業技術総合研究所（旧電子技術総合研究所）の平野らは，IDL を書かなくても Java で作成したプログラムを分散オブジェクトに変えることが容易に出来る HORB（Hirano Object Request Broker）を開発した．

HORB の基本アーキテクチャを図－2 に示す．HORB では，CORBA においてスタブと呼ばれる部分が Proxy（代理）と呼ばれる．リモートオブジェクトを Java で作成し，サーバにおいて HORB でコンパイルすると，自動的にこのオブジェクトの Skeleton と Proxy が作成される．予めユーザ側にはその Proxy を配置し，サーバ側では，常に HORB を起動しておく．これにより，ユーザはクライアントにあるオブジェクトから Proxy と Skeleton を仲介してメッセージを送ることで，



図－2　HORB の基本アーキテクチャ

サーバのリモートオブジェクトを動作させることが出来るのである．

HORB は分散オブジェクト技術の中では知名度は必ずしも高くはないが，オープンソースで無償であり，高速である．土木分野でも，景観イメージの遠隔地での同期表示システム[7]の開発に使用された．本研究でも HORB を使用することとした．

3.　オブジェクト指向技術による ObjFEM

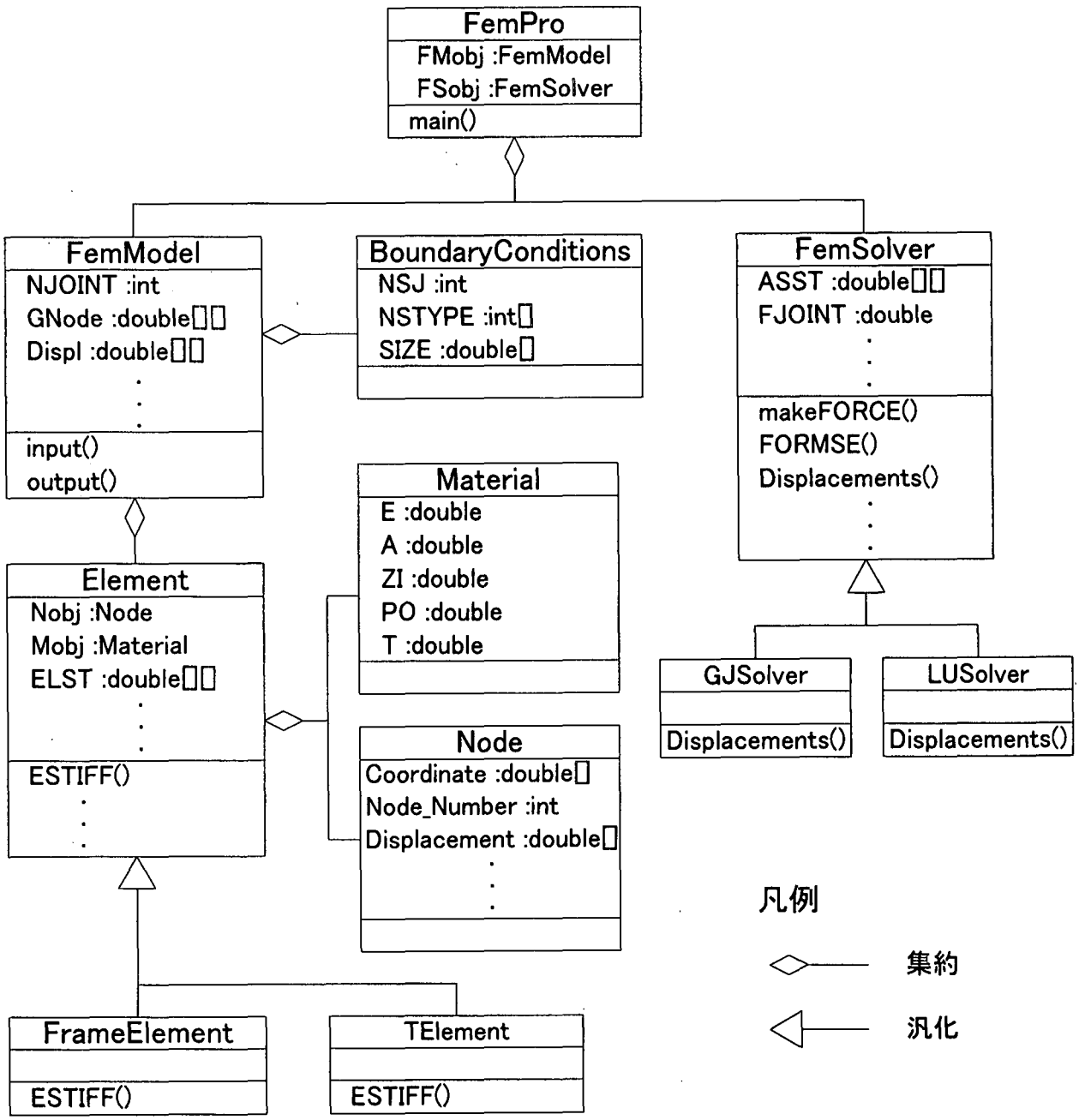
本研究では，まずオブジェクト指向技術により，単体マシン上で動作する FEM コード ObjFEM を開発した．UML によって表現した ObjFEM のクラス図を図－3 に示す．

メインのクラスは FemPro であり，メソッドには main()がある．この下に，“part_of”（集約）

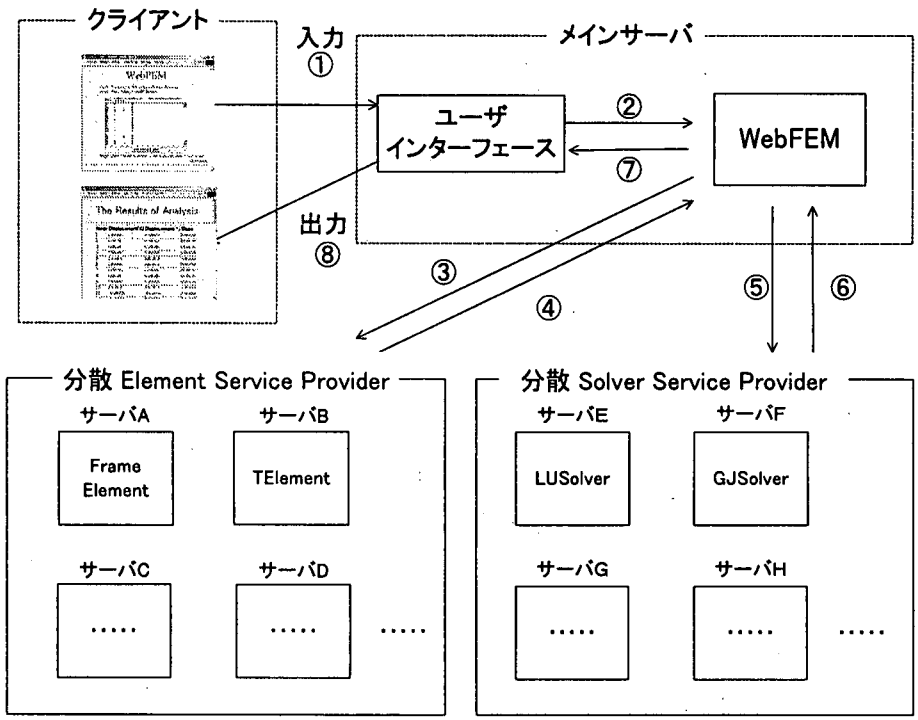
の関係にある2つのクラス FemModel と FemSolver がある．FemModel では，入力データを読取り，FemSolver において他のクラスとのやり取りにより全体剛性マトリクスを作成し，連立方程式を解いて変位を求め，FemModel において計算結果を出力する．

FemModel の下には Element のクラスと境界条件の BoundaryConditions があり，Element の下には"is_a"(汎化)の関係にある FrameElement と TElement の2つのクラスがある．FrameElement はフレーム要素，TElement は2次元弾性三角形要素である．これらは"is_a"の関係にあることから，クラス Element から属性とメソッドは継承されるので，異なる部分だけコーディングすれば良い．また，4角形要素やアイソパラメトリック要素，あるいはもっと高度な要素を開発した場合は，同様に Element クラスの下に設定すればよい．

FemSolver クラスの下にも，"is_a"の関係にあるクラス GJSolver と LUSolver がある．GJSolver は Gauss-Jordan の消去法により連立一次方程式を解くもので，機能的にはベーシック



図－3　ObjFEM のクラス図



図－4　要素とソルバを分散させたシステム構成

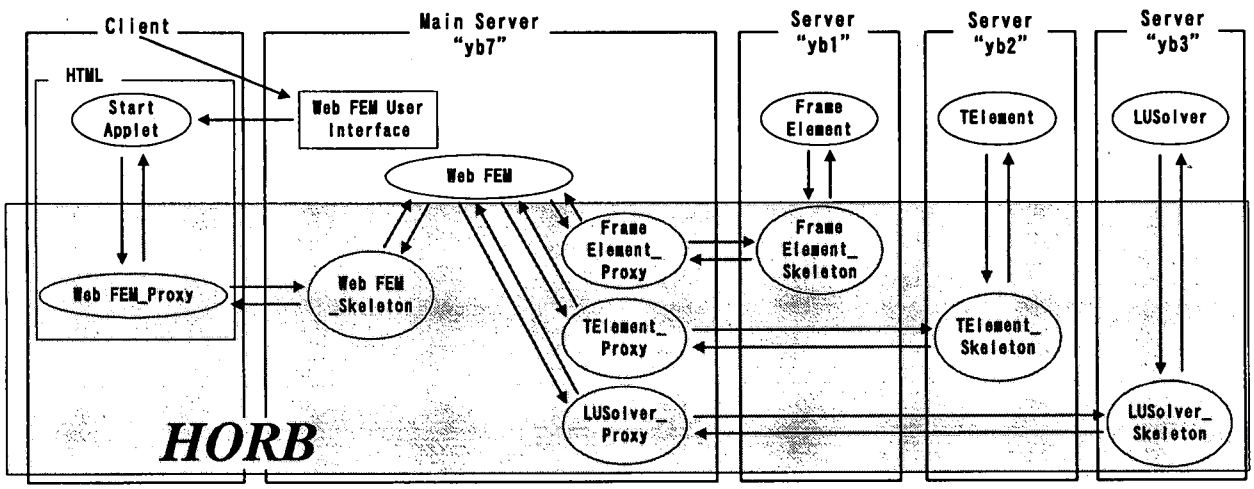
である．一方，LUSolver は，LU 分解法によるもので，機能的には前者より高度である．他にも，連立方程式の解法は多くの研究者により提案されており，そうした解法によるクラスを増やしてライブラリ化することは可能である．

4.　分散オブジェクト技術による WebFEM

協調的に FEM コードを開発する場合や，分担開発者が手元のコンピュータで常にコードをメンテナンスしていきたいといった場合を想定して，分散オブジェクト技術により，要素のクラスとソルバのクラスをメインサーバとは別のサーバ群に分散配置して解析出来る環境を構築した．図－4 にそのシステム構成を示し，以下に，解析の流れを記す．

ユーザは，①クライアントマシン上にある JavaApplet で作成された入力ウィンドウからメインサーバのユーザインターフェースにアクセスし，②WebFEM を起動する．③ユーザの入力データの中で，要素の種類が明示されているので，そのデータにより対応する要素を有する分散 Element Service Provider 内のサーバにアクセスして要素剛性マトリクスを作成して，④WebFEM に返す．結果をもとに全体剛性マトリクスを作成する．⑤次に，ユーザが選択した連立方程式の解法により，分散 Solver Service Provider 内の対応するソルバにアクセスし，⑥解いた結果をメインサーバに返す．WebFEM で出力結果をまとめて，⑦ユーザインターフェースを介して，⑧クライアントに結果を表示する．

図－5 に，Proxy や Skeleton が具体的な例において，どのような位置にあり，データがどう流れ



図－5　HORB による WebFEM の動作状況

るかを示す．この例では，フレーム要素と三角形要素の両方が使用され，ソルバにはLU分解法が選択されている．

HORB を使用することにより，Proxy や Skeleton のコーディングをする必要はないが，実際にどのようなコードが自動的に作成されるのかを示す．図－6に WebFEM.java のソースコードを，図－7に HORB によって自動的に生成された WebFEM_Proxy.java の一部を示す．Proxy が難解であることがわかる．

FEM 解析において，計算時間がかかるのはソルバ部分であるから複数のサーバマシンに同じソルバを置いておけば，もし，他のユーザが既に使いたいソルバを使用中であれば，空いているマシンを使えば負荷分散が図られ，効率的に解析が実行できる．本研究では，LUSolver を2台のサーバにインストールし，1台が計算をしている間にもう1つのジョブを与えたところ，別のサーバを探してジョブを実行することが出来るように WebFEM を設定した．

5. WebFEM の使用例

本研究で開発した WebFEM を使用して，簡単な2次元静的弾性有限要素解析を実施した事例を以下に記す．解析モデルは，図－8に示すような2層のフレームが地盤に立っており，そのフレームには片持ち梁状のトラスが2層目に取り付けてあるものである．各要素の物性値を表－1に示す．

まず，ユーザは WebFEM への入力データ（text file）をあらかじめプリプロセッサ等により作成しておく．サーバにおいては，HORB および Apache[8]が常に動作しており，WWW に接続されている．クライアントも WWW に接続されており，ユーザはインターネットブラウザからサーバ上の URL（http://**.yb7.**/StartApplet.html）にアクセスする．（注：**はセキュリティのため伏せてある．）

ユーザは，StartApplet.java（図－9）によって作成される図－10に示すような入力画面のテキストエリアに先の入力データを入力し，“Start Analysis”ボタンをクリックする．すると，入力データがメインサーバ上の WebFEM に送られ，入力データに基づき WebFEM が必要な動作をそれぞれ適当なサーバへリクエストし FEM 解析が行われる．この解析結果を元に，解析結果を表示する HTML ファイルが作成される．

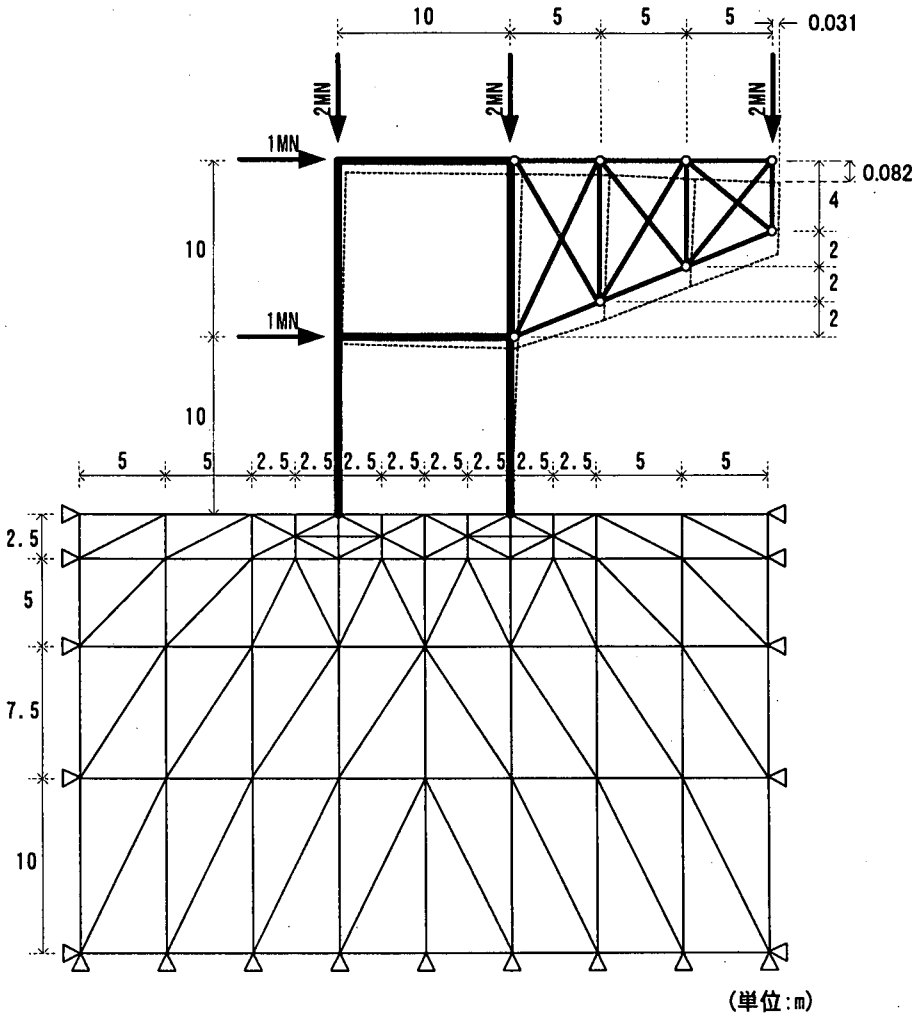
以上の動作の終了後，ユーザがアクセスしている Web ページ上のテキストエリアに“finish”というメッセージが表示されるので，図－10の“show result”というリンクフィールドをクリックすると，計算結果が図－11のように表示される．こうして得られた解析結果を，適当なポストプロセッサに入力すると，ユーザは図－8のような変位図を得る事が可能である．

```
import java.io.*;
import java.text.*;

class WebFEM {
    static FemModel FMobj;
    static FemSolver Sobj;
    public static void start( String s ) {

        makeTextFile( s );

        FMobj = new FemModel();
        Sobj = new FemSolver();
        FMobj.input();
        Sobj.solve( FMobj );

        makeHtmlFile( FMobj.NJOINT );
        Delete();
    }

    static void makeTextFile( String s ) {
        try {
            PrintWriter pw = new PrintWriter(new FileWriter("data.txt"));
            pw.print( s );
            pw.close();
        } catch ( Exception e ) {
            System.out.println( e.getMessage() );
    }
```

図－6　WebFEM.java のソースコード

```
public  class WebFEM_Proxy  implements horb.orb.Proxy {
  protected short classNo;
  public final static short major = 0;
  public final static short minor = 0;
  public short _getMajorVersion() { return major;}
  public short _getMinorVersion() { return minor;}
  protected HorbURL url;
  private transient IOCI ioci;
  public IOCI _getIOCI() { return ioci; }
  public void _setIOCI(IOCI ioci) { this.ioci = ioci; }
  public HorbURL _getObjectURL() { return url; }
  public WebFEM_Proxy() {
    classNo = 1;
  }
  public synchronized void _connect(HorbURL url, String user, String passwd)
    try {
      if (ioci == null)
        ioci = (HORB.getFactory(url.getProtocol())).createIOCI();
    } catch (Exception e) {
      throw new HORBException("can't create IOCI: "+e);
    }
    this.url = ioci.connect(url, "WebFEM",major, minor, user, passwd, horb.or
  }
  public synchronized void _release() {
    ioci.releaseObject();
    ioci = null;
  }
```

図－7　WebFEM_Proxy.java のソースコード



図－８　地盤，フレームおよびトラス構造の FEM 解析モデルおよび解析結果（変位）

表－１　各要素の物性値

	弾性係数(N/m²)	断面積(m²)	ポアソン比	断面2次モーメント(m⁴)
:三角形要素	2000	—	0.25	—
:フレーム要素	20000	0.05	—	0.8
:トラス要素	200000	0.04	—	—

6.　おわりに

本研究では，大規模化・複雑化している FEM コードの開発・メンテナンスの効率化，要素やソルバの協調的な開発環境の開発，１台のコンピュータに負荷を集中させず分散させること，およびユーザが WWW 上で高度な FEM 解析が出来るようにすることを目的に，分散オブジェクト技術を利用した有限要素解析システム WebFEM を開発した．

本システムの開発にあたっては，まず，オブジェクト指向プログラミング言語 Java を用いて FEM 解析ソフトの作成を行い，次にその FEM 解析ソフトを HORB の使用により各コンピュータに分散し，Web を通じて使用可能とした．

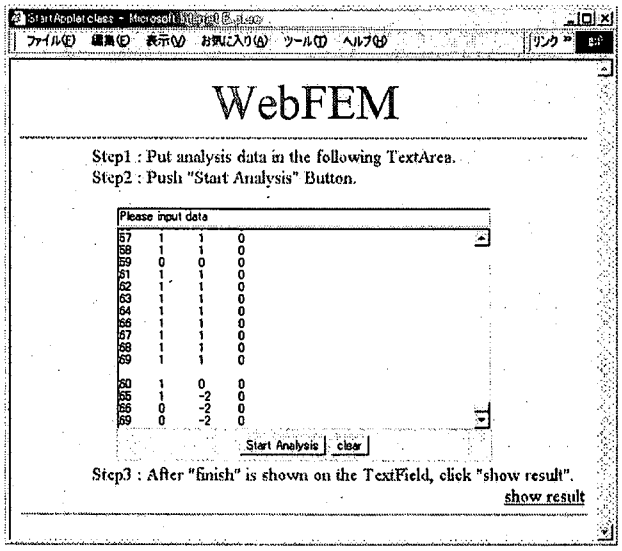
ユーザはあらかじめ解析システムへの入力データを作成し，Web ブラウザを通じて Web サーバにアクセスすることで，FEM 解析を容易に行う事が出来る．クライアントが FEM 解析を行うために最低限必要なものは Web ブラウザのみである．FEM 解析ソフトウェアのバージョンアップ等の作業もサーバ側で行われるため，ユーザ側では，それらの作業に関する時間・労力等のコストを大幅に低減させることが可能だと思われる．

今後の課題としては，本研究で開発した FEM 解析コードをより高度化させること，および実際に協調的に要素やソルバの開発を行う別組織の研究者達と協同作業を行うことが考えられる．

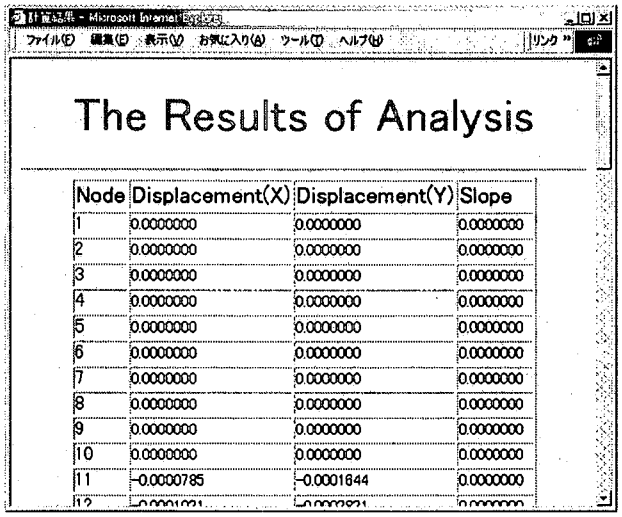
さらに，Web 上で解析モデルを作成し，その解析モデルから直接解析を行うためのプリプロセッサの開発，解析結果を可視化するポストプロセッサの開発，および並列分散処理によるさらなる

```
class StartApplet extends Applet {
    TextField tf;
    TextArea ta;
    String data;
    Button bt1;
    Button bt2;
    Panel p;

    public void init() {
        setLayout( new BorderLayout() );
        add( "Center", ta = new TextArea(20,20) );
        add( "North", tf = new TextField("",20) );
        add( "South", p = new Panel() );
        p.add( bt1 = new Button("Start Analysis") );
        p.add( bt2 = new Button("clear") );
        tf.setText( "Please input data" );

        bt1.addActionListener( new ActionListener() {
            public void actionPerformed( ActionEvent evt ) {
                try {
                    WebFEM_Proxy WP = new WebFEM_Proxy( "horb://**.yb7.**" );
                    data = ta.getText();
                    tf.setText( "Calculating....." );
                    WP.start( data );

                    tf.setText( "finish" );
                } catch ( Exception e ) {
```

図－９　StartApplet.java のソースコード



図－１０　WebFEM の入力画面



Node	Displacement(X)	Displacement(Y)	Slope
1	0.0000000	0.0000000	0.0000000
2	0.0000000	0.0000000	0.0000000
3	0.0000000	0.0000000	0.0000000
4	0.0000000	0.0000000	0.0000000
5	0.0000000	0.0000000	0.0000000
6	0.0000000	0.0000000	0.0000000
7	0.0000000	0.0000000	0.0000000
8	0.0000000	0.0000000	0.0000000
9	0.0000000	0.0000000	0.0000000
10	0.0000000	0.0000000	0.0000000
11	-0.0000785	-0.0001644	0.0000000

図－１１　WebFEM の出力結果画面

高速化等が挙げられる．

また，今後こうした WebFEM を適正な課金により利用することができる ASP（Application Service Provider）を設立する方法論を検討していきたいと考えている．

謝辞

本研究を行うにあたり，米国スタンフォード大の Kincho Henry Law 教授と Jun Peng 氏からご助言を頂いた．ここに謝意を表す．

参考文献

1) 矢川元基，関東康祐：オブジェクト指向計算力学入門，培風館，1999.
2) Archer, G., Thewalt, C., and Fenves, G. L.: A New Architecture for Finite Element Analysis, Proceedings of the Third Congress on Computing in Civil Engineering, pp.683-689, 1996.
3) ジョゼフ・シュムラー：独習 UML，翔泳社，2000.
4) Peng, J., McKenna, F., Fenves, G. L., and Law, K. H.: An Open Collaborative Model for Development of Finite Element Program, Proceedings of the Eighth International Conference on Computing in Civil and Building Engineering, pp.1309-1316, 2000.
5) http://horb.etl.go.jp/horb-j/
6) 河込和宏，中村秀男，大野邦夫，飯島正：分散オブジェクトコンピューティング，共立出版，1999.
7) 長舟利雄，森川直洋，浜嶋鉱一郎：JAVA と HORB による景観イメージ比較のための同期表示システム，第 23 回土木情報システム講演集，pp.125-128, 1998.
8) http://www.apache.org/