〔非公開〕

ＴＲ－Ｍ－００２８

感性語による検索を行うデータベースシステム

吉岡　琢　　井上　誠喜
Taku　YOSHIOKA　　Seiki　INOUE

１９９７．９．３０

ＡＴＲ知能映像通信研究所

感性語による検索を行う データベースシステム

奈良先端科学技術大学院大学
吉岡 琢

平成9年9月30日

目 次

1　序論

近年、感性を情報処理に取り入れようとする試みがさかんに行われている。このことは、情報処理の分野において、技術が中心であった時代から技術を使う人間が重視される時代への変化を意味する。

そのような状況の中、感性をアプリケーションのインターフェースに応用する研究が行われている。本実習では、「感性語」と呼ばれる感性を表す単語を入力することによって、それにふさわしいデータをデータベースから検索するシステムを作成した。

感性ではもちろん主観的な要素が大きい。しかし、感性の研究で主にとられているアプローチは、多人数の感性を統計的に処理したデータに基づいたものである。そこで、本実習では感性の主観的な要素が重要であると考え、使っていくうちにユーザーの嗜好を学習して、推論を行うようなシステムを目標とした。

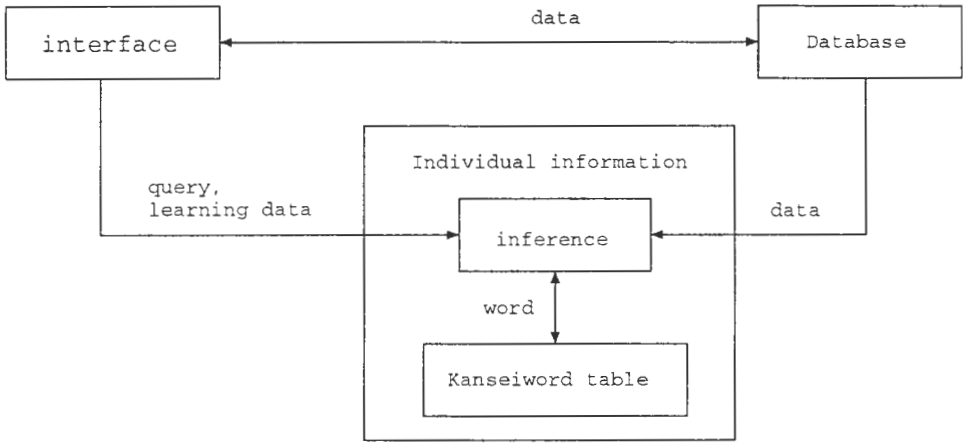


図 1: システムのアーキテクチャ

2 システムの概要

2.1 システムの要素

本実習で作成したシステムは、次の要素から成る (図 1)。

個人情報 個人の（一人の）情報を持つ。推論部と感性語テーブルから成る。システムを使用する前に、ユーザーは自分の個人情報を読み込ませる必要がある。

推論部 特徴ベクトルから感性ベクトルを推論する。

感性語テーブル 感性語と、それに対応する感性ベクトルの成分の番号のテーブル。

データベース 壷のデータが蓄えられている。

インターフェース システムと利用者の間を取り持つ。

特徴ベクトル 検索対象の特徴を表す実数値ベクトル。

感性ベクトル 各成分が、感性語で表される感性の程度に対応する実数値ベクトル。

ユーザーが検索を行うとき、インターフェースを通じて感性ベクトルが入力される。推論部は、データベース中のすべてのデータに対して、特徴ベクトルから感性ベクトルを推論する。そして、ユーザーが入力した感性ベクトルとのユークリッド距離が近いものから順番にソートしたリストを表示する。

もし、表示されたデータに不満があれば、その時点で推論パラメータに修正を加えることができる。その際、推論パラメータを直接入力するのではなく、あるデータとそれにふさわしい感性語の組を入力することによって、間接的に修正を行う。このようにして学習された推論パラメータは、個人情報の一部として保存することができる。次回このシステムを利用するときは、保存された個人情報を読み込むことによって、前回学習した内容を利用することができる。

2.2 検索対象

本実習で作成したシステムにおいて、検索する対象は壷である。ただし、壷の模様は考えず、その形状のみに着目する。データベースは個々の壷のデータを蓄えている。個々の壷のデータは次の3つの要素から成る。

名称 壷の名称を表す。

特徴ベクトル 壷の特徴（本実習では形状）を表すベクトル。

イメージ 壷の様子を表すイメージ (JPEG, GIF)。

これを見れば分かるように、壷のデータには感性情報はいっさい含まれていない。壷に対する感性的な評価を決めるのは特徴ベクトルである。特徴ベクトルの決め方は、システムにとって非常に重要である。壷の特徴を決める要素は、形状は言うに及ばず、テクスチャ、手触り、など様々なものがある。しかし、これらのすべてを表すためには、人間の五感すべてに関するインターフェースが必要であり、現時点ではそれは不可能である。本実習では視覚、それも形状だけに着目しているが、それでも問題は困難である。一般的に、人間がものをみて感性的な判断を下すときに、いつも同じ部分を見ているとは限らない。壷で言えば、あるときは側面の

曲線を、またあるときは壷の口径を見て、判断を下す。したがって、壷の形状に関するあらゆる情報を特徴ベクトルとすればよい結果が期待される。それは、単純に壷の形状を保存しておけばよいのではなく、その形状の持つ、丸みがあるとか細長いというような、形状の特徴を知る必要がある [3]。しかし、これは画像処理の分野になるので、ここでは立ち入らない。

2.3 特徴ベクトル

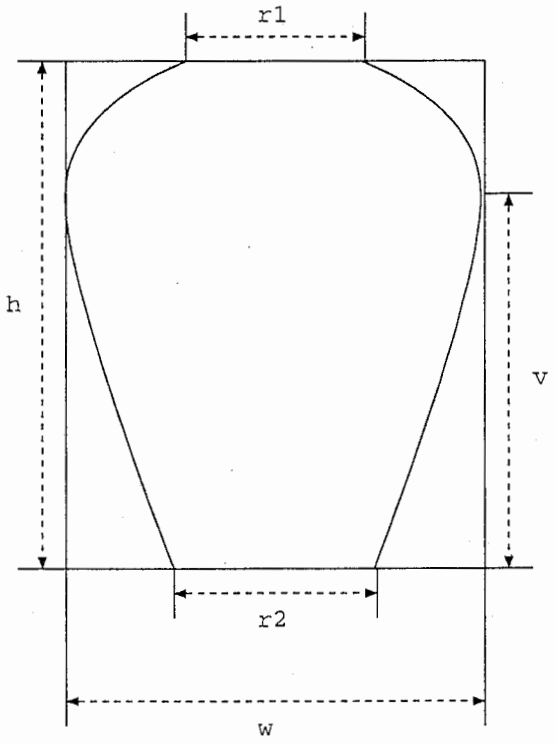
本実習では、単純な次の 4 つの要素を特徴ベクトルの成分とする [1] 。

縦横比 壷の横幅と縦幅の比 (図 2, h/w)。

接点位置比 壷の横幅が最大になる位置の比 (図 2, $(h-v)/w$) 。

口径比 壷の横幅と口径の比 (図 2, $r1/w$)。

底直径比 壷の横幅と底の直径の比 (図 2, $r2/w$)。



Feature Vector = (h/w, (h-v)/v, r1/w, r2/w)

図 2: 特徴ベクトル

3 推論部

3.1 推論部の役割、条件

推論部の役割は、次の2つである。

推論 特徴ベクトルから感性ベクトルへのマッピングを行う。

学習 特徴ベクトルと感性ベクトルの組（学習データ）を受け取り、その評価を学習する。

学習パターンはユーザーによって入力される。したがって、推論部はできるだけ少ない学習パターンで多くの特徴ベクトルを正しく評価できることが望ましい。推論部の実装の方法として、例えば、回帰分析によって感性に影響を与える特徴ベクトルの成分を求める方法や、あるいはもっと高度な方法として、人の認知過程をモデル化してパラメータ推定を行う方法などが考えられる。

本実習ではユーザーを一人に限定しているが、同じ人間でも周囲の環境によって感性的な評価が変わる場合が予想される。このような場合に、学習をどのようにして行えばよいのか、ということも考える必要がある。

3.2 RBF ネットワークによる推論

本実習では、推論部として、Radial Basis Function(RBF) ネットワーク [4] を用いた。このネットワークはベクトル$\boldsymbol{\xi}$を入力として、$\mathbf{O}$を出力する。式は次のようになる。

$$
\begin{aligned}
O_k &= \sum_j w_{jk} g_j(\boldsymbol{\xi}) \\
g_j(\boldsymbol{\xi}) &= \frac{\exp\{-|\boldsymbol{\xi} - \boldsymbol{\mu}_j|^2/\alpha\sigma_j^2\}}{\sum_l \exp\{-|\boldsymbol{\xi} - \boldsymbol{\mu}_l|^2/\alpha\sigma_l^2\}}
\end{aligned}
$$

ここで、関数$g_j(\boldsymbol{\xi})$をRadial Basis Functionと呼ぶ。このネットワークは、特徴ベクトル空間を各$\boldsymbol{\mu}_j$を中心とするクラスタに分割する。つまり、$\boldsymbol{\mu}_j$に近い特徴ベクトルは、$\mathbf{w_j}$に近い感性ベクトルを出力する。すなわち、近い形状を持つ壷は近い感性的な特徴を持つと推論する。しかし、人間の感性を表すためには、このような単純なモデルでは不十分であろう。この部分はシステムにとって最も重要な部分であり、改善する必要がある。

学習は単純に、特徴ベクトルを$\boldsymbol{\mu}_j$、それに対する感性ベクトルを$\mathbf{w_j}$として、RBFを一つ加えることによって行う。同じ特徴ベクトルに関して異なった評価が提示された場合、以前の評価を無視する。σ_jは、一番近くのRBFとの基底間の距離とする。また、$\alpha = 0.5$として、隣接するクラスタの影響が少なくなるようにした。

4 実装

4.1 クラス

本実習では、上述したような感性語によるデータベース検索システムを実装した。プログラミング言語にはJavaを用いた。この章では、作成したクラスについて解説しながら、同時にシステムの利用法についても説明する。主なクラスを次に示す。

- MainWindow
- WordDialog
- WordListDialog
- DataListDialog
- TuboDialog
- NewDataDialog
- Individual
- Tubo

以下、これらのクラスについて説明する。

MainWindow このクラスはメインウィンドウを表す。また、main() メソッドを含んでおり、アプリケーションはここから起動する。メインウィンドウは4つのメニューを持つ。以下、これらのメニューについて説明する。

- "search" このメニューからは、検索を行うことができる。メニューを選択すると、WordDialogが開いて利用者に入力を促す。その後、内部では選択した感性語と壷のデータ（の配列）を引数としてIndividual.search()を呼び出す。そして、その結果をDataListDialogによって表示する。

- “data” このメニューからは、登録されている壷のデータの表示、保存、読み込みを行うことができる。”data - data list” を選択すると、DataListDialogによってデータの一覧が表示される。”data - save” を選択すると、ファイルダイアログが表示され、登録した壷のデータを保存することができる。”data - load” を選択すると、壷のデータを読み込むことができる。
- “individual” このメニューからは、個人情報の保存、読み込みを行うことができる。
- “end” このメニューから、アプリケーションを終了することができる。

WordDialog このクラスは、感性語を入力するダイアログを表す。このダイアログによって、3種類の感性語を同時に指定することができる。各感性語について、テキストを入力するフィールド、感性語が表す感性の程度を表すスクロールバー、そして WordListDialog を開いて、個人情報に登録されている感性語の中の一つを選択することができるボタンが用意されている。

WordListDialog このクラスは、個人情報に登録されている感性語の一覧を表示するダイアログを表す。このダイアログによって、個人情報に登録されている感性語の中の一つを選択することができる。

DataListDialog このクラスは、壷のデータの一覧を表示するダイアログを表す。このダイアログは3つのボタンを持つ。”OK” ボタンを押すとダイアログが閉じる。”select” ボタンを押すことによって、選択されている壷に対する TuboDialog を開くことができる。”new data” ボタンを押すと、NewDataDialogが開いて、新しいデータを追加することができる。

TuboDialog このクラスは壷のデータを表示するダイアログを表す。ダイアログの中央には壷のイメージが表示される。このダイアログは3つのボタンを持つ。”OK” ボタンを押すとダイアログが閉じる。”Kansei” ボタンを押すと WordListDialogが開き、個人情報に登録されている感性語の中の一つを選択させる。すると、この壷の、選択した感性語に関する評価が表示される。“modify” ボタンを押すと、WordDialogが開き、利用者に感性語の入力を促す。そして、入力された感性語をこの壷に対する利用者の評価と考え、個人情報の

推論パラメータを変更する。もし、個人情報に登録されていない感性語が入力されれば、その感性語を個人情報に追加する。

NewDataDialog このクラスは新しいデータを登録するためのダイアログを表す。ダイアログには、登録したい壷の名称、壷のイメージを表すファイル名 (JPEG, GIF)、ノイズを除去した壷の形状データ（レンジデータ）[2] を表すファイル名を入力するためのフィールドがある。後者の 2 つは、"file" ボタンから開くファイルダイアログによって、ファイル名を指定する。また、イメージは"preview" ボタンによってプレビューすることができる。以上のパラメータを指定して"OK" ボタンを押すと、内部では、パラメータに基づいた壷のデータが作成される。

Individual このクラスは個人情報をあらわす。このクラスはデータメンバとして感性語のテーブルと推論用のパラメータを持つ。このクラスは、壷のデータとそれに対する感性語の組を提示する（引数として渡す）ことによって学習する機能を持つ。また、壷のデータと感性語を渡すことによって、壷のデータを感性語による評価の順に並べる機能を持つ。その詳細は、前章で述べた通りである。

Tubo このクラスは個々の壷のデータをあらわす。このクラスはデータメンバとして、名称、イメージのファイル名、壷の形状の特徴ベクトルを持つ。また、このクラスはレンジデータ（ファイル名で指定）を特徴ベクトルに変換するコンストラクタを持つ。

4.2 アプリケーションの使用

使用できるアプリケーションは次の 2 つである。

MainWindow 感性語によるデータベース検索システム。

MakeDatabase サンプルの壺のデータベースを作成する。

MakeIndividual サンプルの個人情報を作成する。

検索システムは Java アプリケーション（アプレットではない）であり、次のようにして起動する。

```
java MainWindow
```

システムは、起動と同時にサンプルの壺のデータベースと個人情報を読み込む。4つのメニューからなるウィンドウが表示される。各メニューが持つ役割については、前述したとおりである。

サンプルの壺のデータベースと個人情報はそれぞれ、MakeDatabase および MakeIndividual によって作成したものである。その使用方法は次の通り。

```
java MakeDatabase(MakeIndividual) [sourcefile] [outputfile]
```

最初の引数は、データを作成する元になるファイルであり、個人情報については感性語の数、その種類、各特徴ベクトルに対する感性評価を指定する。データベースについては、壺の個数、各壺のデータを指定する。2つ目の引数は出力ファイルである。

5　まとめ

本実習では、感性語による検索が可能なデータベースを作成した。しかし、最も肝心な、ユーザーによるテストを行う時間がなかった。このことも含めて、このシステムについて以下のような課題が考えられる。

推論部 本実習では推論部としてRBFネットワークを用いたが、その有効性は検討されていない。有効性の基準はいかに少ない教師データからユーザーの嗜好に近い感性的な推論をするかという事であり、検討の方法は、ユーザーに対する調査が最も適切であると考えられる。例えば、壺のデータをいくつか用意しておき、ユーザーにその壺についての感性的な評価をさせる。これを教師データとして推論部に与える。学習させた推論部を用いて、ある一定の感性語について、教師データの中に含まれていなかった壺のデータベースの検索を行う。指定した感性語とその結果に対するユーザーの感性的な評価とを比較することによって、推論部の有効性を定量的に知ることができる。このような方法を用いて、RBFだけではなく、いくつかの推論方法を比較、検討する必要がある。

インターフェース 壺の評価はユーザーによって入力されるが、そのインターフェースが面倒なものであれば多くの評価を入力するのは敬遠されるだろう。逆に言えば、負担が少ないインターフェースを採用すれば、より多くの評価が得られるかもしれない。

特徴ベクトル このシステムで用いている特徴ベクトルは4つの成分しかないが、もっと多くの要素が感性に関わっていることは明らかである。テクスチャを考えることになると、成分の数はさらに増える。このシステムが個人にあわせて学習を行うシステムである、ということを考えると、感性に関わっている成分とそうでない成分を見つけだすことによって、成分の数を減少させるような仕組みがあることが望ましい。

本実習で作成したシステムは簡素なものであり、特徴ベクトルの成分の少なさだけを考えても、実用にはほど遠い。しかし、このシステムを叩き台にして改良を加えれば、実用的なシステムに近付くと考えている。

参考文献

[1] 中尾「感性データベースに関する調査」ATR 内部資料, 1997

[2] 松下「3 次元部品のテキスチャ&形状の取得に関する研究」ATR 内部資料, 1997

[3] 中村、長尾「微妙なパターンの双方向的解析」文部省科学研究費補助金重点領域研究平成 6 年度成果報告書「感性情報処理の情報学・心理学的研究」pp.37-40, 1995

[4] Hertz, J., Krogh, A., Palmer, R.G., *"INTRODUCTION TO THE THEORY OF NEURAL COMPUTATION"* Addison-Wesley Publishing Company pp.248-250, 1991

A　ディレクトリ構成およびファイルについて

本実習で作成したファイルは全て /home/yoshioka/java/kansei のもとに入っている。拡張子が java のファイルが各クラスのソースファイル、拡張子が class のファイルが Java のクラスファイルである。

individual.src というファイルは、4 章で説明した、データのソースファイルである。このファイルによって、個人情報を作成する。最初の数字は感性語の数、次のいくつかの単語は感性語を表す。感性語の数は、最初に指定した数と一致する必要がある。その次の数字は RBF の数であり、以後、特徴ベクトルとそれに対する感性評価の組が、指定した数だけ続く。

database.src というファイルによってデータベースを作成する。最初にデータの数、その数だけ壺の情報、すなわち名称、イメージのファイル名、特徴ベクトル、が続く。

tree.html ファイルに、実習で作成したクラスがすべて記述されている。各 HTML ファイルは javadoc コマンドによって作成したものである。

data ディレクトリには、壷のイメージファイルがある。image ディレクトリには HTML で用いている GIF イメージがおいてある。

B JDK(Java Development Kit) について

本実習では JDK1.1 を使用したため、それ以前のバージョン（1.0.2 など）では動作しない。このドキュメントを書いている時点で、JDK は /usr/local/jdk1.1.4 にインストールされている。

```
// BarCanvas.java (09/17/1997 Taku Yoshioka)

import java.awt.*;

public class BarCanvas extends Canvas
{
        // Fields

        private double          value;
        private double          max;
        private Color           fore;
        private Color           back;

        // Constructors

        public BarCanvas(double m,double v,Color f,Color b)
        {
                super();

                max=m;
                value=v;
                fore=f;
                back=b;
        }

        // Methods

        public void paint(Graphics g)
        {
                g.setColor(back);
                g.fillRect(0,0,getSize().width,getSize().height);
                g.setColor(fore);
                g.fillRect(0,0,(int)(getSize().width*(value/max)),getSize().height);
        }
}
```

```
// DataLIstDialog.java (09/17/1997 Taku Yoshioka)

import java.awt.*;
import java.awt.event.*;
import java.util.Vector;

public class DataListDialog extends Dialog implements ActionListener
{
        // Fields

        private List            list;
        private Button          ok;
        private Button          select;
        private Button          adddata;
        private Vector          tubodata;
        private Individual      individual;
        private Frame           parent;

        // Constructors

        public DataListDialog(Frame p,Vector td,Individual idv,boolean newdata)
        {
                super(p,"Data",false);

                parent=p;
                tubodata=td;
                individual=idv;

                GridBagConstraints      c=new GridBagConstraints();
                GridBagLayout           g=new GridBagLayout();

                setLayout(g);

                list=new List(10,false);
                c.gridx=0;
                c.gridy=0;
                c.gridwidth=9;
                c.weightx=1;
                c.weighty=0.9;
                g.setConstraints(list,c);
                add(list);

                ok=new Button("   OK   ");
                c.gridx=0;
                c.gridy=1;
                c.gridwidth=3;
                c.weightx=0.333;
                c.weighty=0.1;
                g.setConstraints(ok,c);
                add(ok);

                select=new Button(" select ");
                c.gridx=GridBagConstraints.RELATIVE;
                g.setConstraints(select,c);
                add(select);

                adddata=new Button("new data");
                g.setConstraints(adddata,c);
                add(adddata);

                ok.addActionListener(this);
                select.addActionListener(this);
                adddata.addActionListener(this);

                for(int i=0;i<tubodata.size();i++)
                        list.add(((Tubo)(tubodata.elementAt(i))).name());
```



```
                pack();

                adddata.setEnabled(newdata);
        }

        // Events

        public void actionPerformed(ActionEvent e)
        {
                if(e.getSource()==select)
                {
                        Tubo    tubo=null;
                        int     i;

                        for(i=0;i<tubodata.size();i++)
                        {
                                tubo=(Tubo)(tubodata.elementAt(i));
                                if(tubo.name().compareTo(list.getSelectedItem())==0)
break;
                        }

                        if(i==tubodata.size()) return;

                        else
                        {
                                TuboDialog      dlg=new TuboDialog(parent,tubo,indivi
dual);
                                dlg.setLocation(getLocationOnScreen().x+64,getLocatio
nOnScreen().y+64);
                                dlg.show();
                        }
                }

                else if(e.getSource()==ok)
                        dispose();

                else if(e.getSource()==adddata)
                {
                        NewDataDialog   dlg=new NewDataDialog(parent,tubodata);

                        dlg.setLocation(getLocationOnScreen().x+64,getLocationOnScree
n().y+64);
                        dlg.addData();

                        list.removeAll();

                        for(int i=0;i<tubodata.size();i++)
                                list.add(((Tubo)(tubodata.elementAt(i))).name());

                        repaint();
                }
        }
}
```

```
// EvaluationDialog.java (09/17/1997 Taku Yoshioka)

import java.awt.*;
import java.awt.event.*;

public class EvaluationDialog extends Dialog implements ActionListener
{
        // Fields

        private Button          ok;

        // Constructors

        public EvaluationDialog(Frame parent,String word,double max,double value)
        {
                super(parent,"Evaluation",false);

                GridBagConstraints      c=new GridBagConstraints();
                GridBagLayout           g=new GridBagLayout();

                setLayout(g);

                Label   label=new Label(word);
                c.gridx=0;
                c.gridy=0;
                c.gridwidth=1;
                c.gridheight=1;
                c.weightx=0.1;
                c.weighty=0.5;
                g.setConstraints(label,c);
                add(label);

                BarCanvas       canvas=new BarCanvas(max,value,Color.blue,Color.black
);
                canvas.setSize(256,16);
                c.gridx=GridBagConstraints.RELATIVE;
                c.gridwidth=10;
                c.weightx=0.9;
                g.setConstraints(canvas,c);
                add(canvas);

                ok=new Button("   OK   ");
                c.gridx=0;
                c.gridy=1;
                c.gridwidth=1;
                c.weightx=0.1;
                g.setConstraints(ok,c);
                add(ok);

                pack();

                ok.addActionListener(this);
        }

        // Events

        public void actionPerformed(ActionEvent e)
        {
                if(e.getSource()==ok)
                        dispose();
        }
}
```

```
// IllegalWordException.java (09/10/1997 Taku Yoshioka

public class IllegalWordException extends Exception
{
        // Fields

        private String          word;

        // Constructors

        public IllegalWordException(String w)
        {
                word=w;
        }

        // Methods

        public String word()
        {
                return word;
        }
}
```

```
// ImageCanvas.java (09/16/1997 Taku Yoshioka)

import java.awt.*;

public class ImageCanvas extends Canvas
{
        // Fields

        private Image   image;

        // Constructors

        public ImageCanvas(Image i)
        {
                super();

                image=i;
        }

        // Methods

        public void paint(Graphics g)
        {
                if(image!=null)
                        g.drawImage(image,0,0,getSize().width,getSize().height,this);
        }

        public void setImage(Image img) {image=img;}
}
```

```
// Individual.java (09/10/1997 Taku Yoshioka)

import java.util.*;
import java.io.*;

public class Individual implements Serializable
{
        // Fields

        private double[][]      x;
        private double[]        sigma;
        private double[][]      w;
        private int             numinput;
        private int             numRBF;
        private int             numoutput;

        private Hashtable       words;          // Kansei words

        // Constructors

        public Individual(int n)
        {
                numinput=n;
                numRBF=numoutput=0;

                x=new double[numinput][numRBF];
                w=new double[numRBF][numoutput];
                sigma=new double[numRBF];

                words=new Hashtable();
        }

        // Serialization

        private void writeObject(ObjectOutputStream out)
                                                throws IOException
        {
                out.writeInt(numinput);
                out.writeInt(numRBF);
                out.writeInt(numoutput);

                for(int i=0;i<numinput;i++)
                        for(int j=0;j<numRBF;j++)
                                out.writeDouble(x[i][j]);

                for(int j=0;j<numRBF;j++)
                        for(int k=0;k<numoutput;k++)
                                out.writeDouble(w[j][k]);

                for(int j=0;j<numRBF;j++)
                        out.writeDouble(sigma[j]);

                out.writeObject(words);
        }

        private void readObject(ObjectInputStream in)
                        throws IOException,ClassNotFoundException
        {
                numinput=in.readInt();
                numRBF=in.readInt();
                numoutput=in.readInt();

                x=new double[numinput][numRBF];
                w=new double[numRBF][numoutput];
                sigma=new double[numRBF];
```



```
                for(int i=0;i<numinput;i++)
                        for(int j=0;j<numRBF;j++)
                                x[i][j]=in.readDouble();

                for(int j=0;j<numRBF;j++)
                        for(int k=0;k<numoutput;k++)
                                w[j][k]=in.readDouble();

                for(int j=0;j<numRBF;j++)
                        sigma[j]=in.readDouble();

                words=(Hashtable)(in.readObject());
        }

        // Methods

        public double[] RBFOutput(double[] in)
        {
                double[]        A=new double[numRBF];
                double          sum=0;

                for(int j=0;j<numRBF;j++)
                {
                        double  r=0;

                        for(int i=0;i<numinput;i++)
                                r+=(in[i]-x[i][j])*(in[i]-x[i][j]);

                        A[j]=Math.exp(-r/(0.5*sigma[j]*sigma[j]));
                        sum+=A[j];
                }

                for(int j=0;j<numRBF;j++) A[j]/=sum;

                return A;
        }

        public double[] output(double[] in)
        {
                double[]        output=new double[numoutput];
                double[]        A=RBFOutput(in);

                for(int k=0;k<numoutput;k++)
                {
                        output[k]=0;

                        for(int j=0;j<numRBF;j++)
                                output[k]+=w[j][k]*A[j];
                }

                return output;
        }

        public void addWord(String wd)
        {
                if(words.get(wd)!=null) return;

                double[][]      temp=new double[numRBF][numoutput+1];

                for(int j=0;j<numRBF;j++)
                {
                        for(int k=0;k<numoutput;k++)
                                temp[j][k]=w[j][k];

                        temp[j][numoutput]=0.5;
                }
```

```
        words.put(wd,new Integer(numoutput));
        w=temp;
        numoutput++;
    }

    public void addOutput(double[] pw)
    {
        double[][]      temp=new double[numRBF][numoutput+1];

        for(int j=0;j<numRBF;j++)
        {
            for(int k=0;k<numoutput;k++)
                temp[j][k]=w[j][k];

            temp[j][numoutput]=pw[j];
        }

        w=temp;
        numoutput++;
    }

    public void addRBF(double[] px,double[] pw)
    {
        double[][]      temp1=new double[numinput][numRBF+1];
        double[][]      temp2=new double[numRBF+1][numoutput];
        double[]        temp3=new double[numRBF+1];

        for(int i=0;i<numinput;i++)
        {
            for(int j=0;j<numRBF;j++)
                temp1[i][j]=x[i][j];

            temp1[i][numRBF]=px[i];
        }

        for(int k=0;k<numoutput;k++)
        {
            for(int j=0;j<numRBF;j++)
                temp2[j][k]=w[j][k];

            temp2[numRBF][k]=pw[k];
        }

        for(int j=0;j<numRBF;j++) temp3[j]=sigma[j];

        temp3[numRBF]=1.0;

        x=temp1;
        w=temp2;
        sigma=temp3;
        numRBF++;
    }

    public double evaluate(double[] in,Hashtable aux)
                                throws IllegalWordException
    {
        double[]        out=new double[w[0].length];
        double[]        result=output(in);
        double          v=0;

        for(int i=0;i<out.length;i++) out[i]=-1;

        for(Enumeration e=aux.keys();e.hasMoreElements();)
        {
            String  key=(String)(e.nextElement());
```



```
            out[word2Index(key)]=((Double)(aux.get(key))).doubleValue();
        }

        for(int i=0;i<out.length;i++)
        {
            if(out[i]!=-1) v+=(out[i]-result[i])*(out[i]-result[i]);
        }

        return v;
    }

    public Enumeration words() {return words.keys();}

    public int word2Index(String word) throws IllegalWordException
    {
        Integer         index=(Integer)(words.get(word));

        if(index==null) throw new IllegalWordException(word);
        else return index.intValue();
    }

    public void setSigma()
    {
        for(int j=0;j<numRBF;j++)
        {
            double  min=Double.MAX_VALUE;
            double  temp;

            for(int j2=0;j2<numRBF;j2++)
            {
                if(j2==j) continue;

                temp=RBFdistance(j,j2);

                if(temp<min) min=temp;
            }

            sigma[j]=Math.sqrt(min);
        }
    }

    public double RBFdistance(int j,int j2)
    {
        double  d=0;

        for(int i=0;i<numinput;i++)
            d+=(x[i][j]-x[i][j2])*(x[i][j]-x[i][j2]);

        return d;
    }

    public void modify(double[] in,Hashtable aux)
    {
        for(int j=0;j<numRBF;j++)
        {
            boolean         eq=true;

            for(int i=0;i<numinput;i++)
            {
                if(x[i][j]!=in[i])
                {
                    eq=false;
                    break;
                }
            }
```

```
                        if(eq)
                        {
                                for(Enumeration e=aux.keys();e.hasMoreElements();)
                                {
                                        String  key=(String)(e.nextElement());

                                        try {w[j][word2Index(key)]=((Double)(aux.get(
key))).doubleValue();}

                                        catch(IllegalWordException ex)
                                        {
                                                ex.printStackTrace();

                                                addWord(key);
                                                w[j][numoutput-1]=((Double)(aux.get(k
ey))).doubleValue();
                                        }
                                }

                                return;
                        }
                }

                double[]        pw=new double[numoutput];

                for(int k=0;k<numoutput;k++) pw[k]=0.5;

                for(Enumeration e=aux.keys();e.hasMoreElements();)
                {
                        String  key=(String)(e.nextElement());

                        try{pw[word2Index(key)]=((Double)(aux.get(key))).doubleValue(
);}

                        catch(IllegalWordException ex)
                        {
                                addWord(key);
                                pw=new double[numoutput];
                                pw[numoutput-1]=((Double)(aux.get(key))).doubleValue(
);
                        }
                }

                addRBF(in,pw);
                setSigma();
        }

        public Vector search(Vector tubodata,Hashtable aux)
        {
                Vector  temp1=new Vector();
                Vector  temp2=new Vector();

                try
                {
                        for(int i=0;i<tubodata.size();i++)
                        {
                                Tubo    tubo=(Tubo)(tubodata.elementAt(i));
                                double  d=evaluate(tubo.feature(),aux);
                                int     j;

                                System.out.println(tubo.name()+":"+d);

                                for(j=0;j<temp1.size();j++)
                                {
                                        if(d<((Double)(temp1.elementAt(j))).doubleVal
```



```
ue())
                                                break;
                                }

                                temp1.insertElementAt(new Double(d),j);
                                temp2.insertElementAt(tubo,j);
                        }
                }
                catch(IllegalWordException ex) {ex.printStackTrace();}

                return temp2;
        }
}
```

```
// MainWindow.java (09/16/1997 Taku Yoshioka)

import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import java.io.*;

public class MainWindow extends Frame implements ActionListener
{
        // Fields

        private Menu            searchmenu;
        private Menu            datamenu;
        private Menu            indvmenu;
        private Menu            endmenu;
        private MenuItem[]      searchitem;
        private MenuItem[]      dataitem;
        private MenuItem[]      indvitem;
        private MenuItem[]      enditem;

        private Vector          tubodata;
        private Individual      individual;

        // Constructors

        public MainWindow()
        {
                MenuBar         mb=new MenuBar();

                searchmenu=new Menu("  search  ");
                datamenu=new Menu("   data   ");
                indvmenu=new Menu("individual");
                endmenu=new Menu("   end   ");

                setMenuBar(mb);

                mb.add(searchmenu);
                mb.add(datamenu);
                mb.add(indvmenu);
                mb.add(endmenu);

                searchitem=new MenuItem[1];

                searchitem[0]=new MenuItem("search");
                searchitem[0].addActionListener(this);
                searchmenu.add(searchitem[0]);

                dataitem=new MenuItem[3];

                dataitem[0]=new MenuItem("data list");
                dataitem[0].addActionListener(this);
                datamenu.add(dataitem[0]);

                dataitem[1]=new MenuItem("save");
                dataitem[1].addActionListener(this);
                datamenu.add(dataitem[1]);

                dataitem[2]=new MenuItem("load");
                dataitem[2].addActionListener(this);
                datamenu.add(dataitem[2]);

                indvitem=new MenuItem[2];

                indvitem[0]=new MenuItem("save");
                indvitem[0].addActionListener(this);
```



```
                indvmenu.add(indvitem[0]);

                indvitem[1]=new MenuItem("load");
                indvitem[1].addActionListener(this);
                indvmenu.add(indvitem[1]);

                enditem=new MenuItem[2];

                enditem[0]=new MenuItem("cancel");
                enditem[0].addActionListener(this);
                endmenu.add(enditem[0]);

                enditem[1]=new MenuItem("end");
                enditem[1].addActionListener(this);
                endmenu.add(enditem[1]);

                setSize(384,64);

                init();
        }

        // Events

        public void actionPerformed(ActionEvent e)
        {
                // buttons

                if(e.getSource()==searchitem[0])
                {
                        WordDialog      sd=new WordDialog(this,individual);

                        sd.setLocation(getLocationOnScreen().x+64,getLocationOnScreen
().y+64);

                        if(sd.input()==true)
                        {
                                Hashtable       aux=new Hashtable();
                                String[]        word=sd.getWords();
                                double[]        val=sd.getValue();
                                Vector          temp;

                                for(int i=0;i<3;i++)
                                {
                                        if(word[i].compareTo("")==0) continue;
                                        aux.put(word[i],new Double(val[i]));
                                }

                                // evaluation and sorting

                                temp=individual.search(tubodata,aux);

                                //TuboDialog    dlg1;
                                DataListDialog  dlg2;

                                //dlg1=new TuboDialog(this,(Tubo)(temp.elementAt(0)),
individual);
                                //dlg1.setLocation(getLocationOnScreen().x+64,
                                //                 getLocationOnScreen().y+64);
                                //dlg1.show();

                                dlg2=new DataListDialog(this,temp,individual,false);
                                dlg2.setLocation(getLocationOnScreen().x+64,getLocati
onOnScreen().y+64);
                                dlg2.show();
                        }
                }
```

```
                else if(e.getSource()==dataitem[0])
                {
                        DataListDialog  dlg=new DataListDialog(this,tubodata,individu
al,true);

                        dlg.setLocation(getLocationOnScreen().x+64,getLocationOnScree
n().y+64);
                        dlg.show();
                }

                else if(e.getSource()==dataitem[1])
                {
                        try
                        {
                                FileDialog              fd;

                                fd=new FileDialog(this,"",FileDialog.SAVE);
                                fd.setLocation(getLocationOnScreen().x+64,getLocation
OnScreen().y+64);
                                fd.show();

                                if(fd.getFile()==null) return;

                                (new ObjectOutputStream(new FileOutputStream(fd.getDi
rectory()+
                                                                fd.getFile()))).write
Object(tubodata);
                        }
                        catch(IOException ex) {ex.printStackTrace();}
                }

                else if(e.getSource()==dataitem[2])
                {
                        try
                        {
                                FileDialog              fd;

                                fd=new FileDialog(this,"",FileDialog.LOAD);
                                fd.setLocation(getLocationOnScreen().x+64,getLocation
OnScreen().y+64);
                                fd.show();

                                if(fd.getFile()==null) return;

                                tubodata=(Vector)((new ObjectInputStream(
                                        new FileInputStream(fd.getDirectory()
                                                        +fd.getFile()))).readObject()
);
                        }
                        catch(IOException ex) {ex.printStackTrace();}
                        catch(ClassNotFoundException ex) {ex.printStackTrace();}
                }

                else if(e.getSource()==indvitem[0])
                {
                        try
                        {
                                FileDialog              fd;

                                fd=new FileDialog(this,"",FileDialog.SAVE);
                                fd.setLocation(getLocationOnScreen().x+64,getLocation
OnScreen().y+64);
                                fd.show();

                                if(fd.getFile()==null) return;
```



```
                                (new ObjectOutputStream(new FileOutputStream(fd.getDi
rectory()+
                                                                fd.getFile()))).write
Object(individual);
                        }
                        catch(IOException ex) {ex.printStackTrace();}
                }

                else if(e.getSource()==indvitem[1])
                {
                        try
                        {
                                FileDialog              fd;

                                fd=new FileDialog(this,"",FileDialog.LOAD);
                                fd.setLocation(getLocationOnScreen().x+64,getLocation
OnScreen().y+64);
                                fd.show();

                                if(fd.getFile()==null) return;

                                individual=(Individual)((new ObjectInputStream(
                                        new FileInputStream(fd.getDirectory()+
                                                        fd.getFile()))).readObject())
;
                        }
                        catch(IOException ex) {ex.printStackTrace();}
                        catch(ClassNotFoundException ex) {ex.printStackTrace();}
                }

                else if(e.getSource()==enditem[1]) System.exit(0);
        }

        // Methods

        private void init()
        {
                try
                {
                        ObjectInputStream       ois;

                        ois=new ObjectInputStream(new FileInputStream(
                                        "/home/yoshioka/java/kansei/data/tubodata"));
                        tubodata=(Vector)(ois.readObject());

                        ois=new ObjectInputStream(new FileInputStream(
                                        "/home/yoshioka/java/kansei/data/individual")
);
                        individual=(Individual)(ois.readObject());
                }
                catch(Exception e) {e.printStackTrace();}
        }

        public static void main(String[] args)
        {
                MainWindow      mw=new MainWindow();

                mw.setTitle("Kansei Search Application");
                mw.show();
        }
}
```

```
// MakeDatabase.java (09/25/1997 Taku Yoshioka)

import java.io.*;
import java.util.Vector;

public class MakeDatabase
{
        public static void main(String[] args)
        {
                if(args.length<2)
                        System.out.println("java MakeDatabase sourcefile outputfile")
;

                try
                {
                        Vector          tubodata=new Vector();
                        StreamTokenizer stk=new StreamTokenizer(new FileReader(args[0
]));
                        int             numdata;
                        double[]        px;
                        String          filename;

                        ObjectOutputStream      tubooos=new ObjectOutputStream
                                                        (new FileOutputStream(args[1]
));

                        stk.nextToken();
                        numdata=(int)(stk.nval);

                        px=new double[4];

                        for(int i=0;i<numdata;i++)
                        {
                                System.out.println(i);

                                String          name;
                                String          imgname;

                                stk.nextToken();name=stk.sval;
                                System.out.println(stk.sval);
                                stk.nextToken();imgname=stk.sval;
                                System.out.println(stk.sval);

                                for(int j=0;j<4;j++)
                                {
                                        stk.nextToken();
                                        px[j]=stk.nval;
                                }

                                tubodata.addElement(new Tubo(name,imgname,px));
                        }

                        tubooos.writeObject(tubodata);
                }
                catch(Exception e) {e.printStackTrace();}
        }
}
```

```
// MakeIndividual.java (09/25/1997 Taku Yoshioka)

import java.io.*;
import java.util.Vector;

public class MakeIndividual
{
        public static void main(String[] args)
        {
                if(args.length<2)
                        System.out.println("java MakeIndividual sourcefile outputfile
");

                try
                {
                        Individual      idv=new Individual(4);
                        StreamTokenizer stk=new StreamTokenizer(new FileReader(args[0
]));
                        int             numword;
                        int             numdata;
                        double[]        pw;
                        double[]        px;

                        ObjectOutputStream      idvoos=new ObjectOutputStream
                                                        (new FileOutputStream(args[1]
));

                        stk.nextToken();
                        numword=(int)(stk.nval);

                        for(int i=0;i<numword;i++)
                        {
                                stk.nextToken();
                                System.out.println(stk.sval);
                                idv.addWord(stk.sval);
                        }

                        stk.nextToken();
                        numdata=(int)(stk.nval);

                        px=new double[4];
                        pw=new double[numword];

                        for(int i=0;i<numdata;i++)
                        {
                                System.out.println(i);

                                for(int j=0;j<4;j++)
                                {
                                        stk.nextToken();
                                        px[j]=stk.nval;
                                }

                                for(int j=0;j<numword;j++)
                                        pw[j]=0.5;

                                while(true)
                                {
                                        String  word;

                                        stk.nextToken();
                                        word=stk.sval;
                                        if(word.compareTo("end")==0) break;

                                        stk.nextToken();
                                        pw[idv.word2Index(word)]=stk.nval;
```



```
                                }

                                idv.addRBF(px,pw);
                        }

                        idv.setSigma();
                        idvoos.writeObject(idv);
                }
                catch(Exception e) {e.printStackTrace();}
        }
}
```

```
// NewDataDaialog.java (09/11/1997 Taku Yoshioka)

import java.awt.*;
import java.awt.event.*;
import java.util.*;

public class NewDataDialog extends Dialog implements ActionListener
{
        // Fields

        private Button                  ok;
        private Button                  cancel;
        private Button                  imgfile;
        private Button                  imgview;
        private Button                  srcfile;
        private TextField               name;
        private TextField               imgname;
        private TextField               srcname;
        private boolean                 isok;
        private Frame                   parent;

        private Vector                  tubodata;

        // Constructors

        public NewDataDialog(Frame p,Vector td)
        {
                super(p,"Data",true);

                parent=p;
                tubodata=td;

                isok=false;

                Button                  button;
                Label                   label;
                GridBagConstraints      c=new GridBagConstraints();
                GridBagLayout           g=new GridBagLayout();

                setLayout(g);

                label=new Label("name");
                c.gridx=0;
                c.gridy=0;
                c.gridwidth=1;
                c.weightx=0.1;
                c.weighty=0.2;
                g.setConstraints(label,c);
                add(label);

                name=new TextField(32);
                c.gridx=GridBagConstraints.RELATIVE;
                c.gridwidth=6;
                c.weightx=0.6;
                c.fill=GridBagConstraints.HORIZONTAL;
                g.setConstraints(name,c);
                add(name);

                label=new Label("image");
                c.gridx=0;
                c.gridy=1;
                c.gridwidth=1;
                c.weightx=0.1;
                c.weighty=0.2;
                g.setConstraints(label,c);
                add(label);
```



```
                imgname=new TextField(32);
                imgname.setEditable(false);
                c.gridx=GridBagConstraints.RELATIVE;
                c.gridwidth=6;
                c.weightx=0.6;
                c.fill=GridBagConstraints.HORIZONTAL;
                g.setConstraints(imgname,c);
                add(imgname);

                imgfile=new Button("  file  ");
                c.gridwidth=1;
                c.weightx=0.15;
                c.fill=GridBagConstraints.NONE;
                g.setConstraints(imgfile,c);
                add(imgfile);

                imgview=new Button("preview");
                g.setConstraints(imgview,c);
                add(imgview);

                label=new Label("source");
                c.gridx=0;
                c.gridy=2;
                c.gridwidth=1;
                c.weightx=0.1;
                g.setConstraints(label,c);
                add(label);

                srcname=new TextField(32);
                srcname.setEditable(false);
                c.gridx=GridBagConstraints.RELATIVE;
                c.gridwidth=6;
                c.weightx=0.6;
                c.fill=GridBagConstraints.HORIZONTAL;
                g.setConstraints(srcname,c);
                add(srcname);

                srcfile=new Button("  file  ");
                c.weightx=0.15;
                c.gridwidth=1;
                c.fill=GridBagConstraints.NONE;
                g.setConstraints(srcfile,c);
                add(srcfile);

                ok=new Button("   OK   ");
                c.gridx=1;
                c.gridy=4;
                c.weightx=0.7;
                g.setConstraints(ok,c);
                add(ok);
                ok.addActionListener(this);

                cancel=new Button(" cancel ");
                c.gridx=GridBagConstraints.RELATIVE;
                c.weightx=0.3;
                g.setConstraints(cancel,c);
                add(cancel);
                cancel.addActionListener(this);

                pack();

                imgfile.addActionListener(this);
                imgview.addActionListener(this);
                srcfile.addActionListener(this);
        }
```

```
        // Events

        public void actionPerformed(ActionEvent e)
        {
                if(e.getSource()==imgfile)
                {
                        FileDialog      fd;

                        fd=new FileDialog(parent,"Image File",FileDialog.LOAD);
                        fd.setLocation(getLocationOnScreen().x+64,getLocationOnScreen
().y+64);
                        fd.show();

                        imgname.setText(fd.getDirectory()+fd.getFile());
                }

                else if(e.getSource()==imgview)
                {
                        ImageDialog     dlg=new ImageDialog(parent,getToolkit().
                                                getImage(imgname.getText()));

                        dlg.setLocation(getLocationOnScreen().x+64,getLocationOnScree
n().y+64);
                        dlg.show();
                }

                else if(e.getSource()==srcfile)
                {
                        FileDialog      fd;

                        fd=new FileDialog(parent,"Source File",FileDialog.LOAD);
                        fd.setLocation(getLocationOnScreen().x+64,getLocationOnScreen
().y+64);
                        fd.show();

                        srcname.setText(fd.getDirectory()+fd.getFile());
                }

                else if(e.getSource()==ok)
                {
                        isok=true;
                        dispose();
                }

                else if(e.getSource()==cancel)
                {
                        isok=false;
                        dispose();
                }
        }

        // Methods

        public void addData()
        {
                show();

                if(isok==true)
                        tubodata.addElement(new Tubo(name.getText()
                                ,imgname.getText(),srcname.getText()));
        }
}
```

```
// Test.java (09/22/1997 Taku Yoshioka)

import java.awt.*;
import java.awt.event.*;

public class Test extends java.applet.Applet implements MouseListener
{
	// Fields

	private Individual	rbf;
	private double[][]	x;
	private double[][]	w;
	private int		n;

	// Constructors

	public Test() {}

	// Methods

	public void init()
	{
		rbf=new Individual(2);

		double[]	ww=new double[0];

		rbf.addOutput(ww);

		n=(new Integer(getParameter("numRBF"))).intValue();

		x=new double[n][2];
		w=new double[n][1];

		for(int i=0;i<n;i++)
		{
			x[i][0]=(new Double(getParameter("x"+(i+1)))).doubleValue();
			x[i][1]=(new Double(getParameter("y"+(i+1)))).doubleValue();
			w[i][0]=(new Double(getParameter("w"+(i+1)))).doubleValue();
			//double	sigma=(new Double(getParameter("sigma"+(i+1))
)).doubleValue();

			rbf.addRBF(x[i],w[i]);
		}

		rbf.setSigma();

		addMouseListener(this);
	}

	public void paint(Graphics g)
	{
		double[]	r=new double[2];

		for(int y=0;y<96;y++)
			for(int x=0;x<96;x++)
			{
				r[0]=(x-48)/24.0;
				r[1]=(y-48)/24.0;

				double[]	o=rbf.output(r);
				int		c=(int)(o[0]*255);

				if(c<0) c=0;
				if(c>255) c=255;

				g.setColor(new Color(0,0,c));
```



```
				g.fillRect(x*4,y*4,4,4);
			}

		g.setColor(Color.red);

		for(int i=0;i<n;i++)
			g.fillRect((int)(x[i][0]*96+192-2),(int)(x[i][1]*96+192-2),4,
4);
	}

	// Events

	public void mouseClicked(MouseEvent e)
	{
		double[]	r=new double[2];
		double[]	o;

		r[0]=(e.getX()-192)/96.0;
		r[1]=(e.getY()-192)/96.0;

		o=rbf.output(r);
		System.out.println(o[0]);
	}

	public void mousePressed(MouseEvent e) {}

	public void mouseReleased(MouseEvent e) {}

	public void mouseEntered(MouseEvent e) {}

	public void mouseExited(MouseEvent e) {}
}
```

```
// Tubo.java (09/12/1997 Taku Yoshioka)

import java.applet.Applet;
import java.io.*;
import java.awt.*;
import java.net.URL;

public class Tubo implements Serializable
{
        // Fields

        private double[]        feature;
        private Image           image;
        private String          imagename;
        private String          name;

        // Constructors

        public Tubo(String n,String imgname,double[] src)
        {
                name=new String(n);
                imagename=new String(imgname);
                feature=new double[4];

                for(int i=0;i<4;i++)
                        feature[i]=src[i];
        }

        public Tubo(String n,String imgname,String srcname)
        {
                name=new String(n);
                imagename=new String(imgname);
                feature=new double[4];

                int             count;
                int             prev;
                int             prevwidth;

                int             upper;
                int             lower;
                int             max;
                int             upperwidth;
                int             lowerwidth;
                int             maxwidth;

                try{
                        StreamTokenizer stk=new StreamTokenizer(new InputStreamReader
(
                                                                new FileInputStream(s
rcname)));

                        while(true)
                        {
                                stk.nextToken();
                                count=(int)(stk.nval);
                                stk.nextToken();
                                if(stk.nval>0) break;
                        }

                        prev=max=lower=count;
                        prevwidth=maxwidth=lowerwidth=(int)(stk.nval);

                        while(true)
                        {
                                stk.nextToken();
                                count=(int)(stk.nval);
```



```
                                stk.nextToken();
                                if(stk.nval<=0) break;
                                if(stk.nval>maxwidth)
                                {
                                        maxwidth=(int)(stk.nval);
                                        max=count;
                                }

                                prev=count;
                                prevwidth=(int)(stk.nval);
                        }

                        upper=prev;
                        upperwidth=prevwidth;

                        feature[0]=(upper-lower)/(double)maxwidth*100;
                        if(max!=lower) feature[1]=(upper-max)/(double)(max-lower);
                        else feature[1]=1000;
                        feature[2]=upperwidth/(double)maxwidth;
                        feature[3]=lowerwidth/(double)maxwidth;

                        System.out.println("upper     :"+upper);
                        System.out.println("lower     :"+lower);
                        System.out.println("max       :"+max);
                        System.out.println("upperwidth:"+upperwidth);
                        System.out.println("lowerwidth:"+lowerwidth);
                        System.out.println("maxwidth  :"+maxwidth);
                        System.out.println(feature[0]);
                        System.out.println(feature[1]);
                        System.out.println(feature[2]);
                        System.out.println(feature[3]);
                }
                catch(Exception e) {e.printStackTrace();}
        }

        // Serialization

        private void writeObject(ObjectOutputStream out)
                                                throws IOException
        {
                for(int i=0;i<4;i++)
                        out.writeDouble(feature[i]);

                out.writeObject(imagename);
                out.writeObject(name);
        }

        private void readObject(ObjectInputStream in)
                                        throws IOException, ClassNotFoundException
        {
                feature=new double[4];

                for(int i=0;i<4;i++)
                        feature[i]=in.readDouble();

                imagename=(String)(in.readObject());
                name=(String)(in.readObject());
        }

        // Methods

        public Image image() {return (new Frame()).getToolkit().getImage(imagename);}

        public double[] feature() {return feature;}

        public String name() {return name;}
```

```
}
```

```
// WordDaialog.java (09/09/1997 Taku Yoshioka)

import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;
import java.util.*;

public class WordDialog extends Dialog implements ActionListener
{
        // Fields

        private Button                  ok;
        private Button                  cancel;
        private TextField[]             text;
        //private Checkbox[]            chkbox;
        private Scrollbar[]             scrbar;
        private Button[]                button;
        private boolean                 isok;
        private Individual              individual;
        private Frame                   parent;

        // Constructors

        public WordDialog(Frame p,Individual idv)
        {
                super(p,"Word List",true);

                parent=p;
                individual=idv;

                isok=false;

                text=new TextField[3];
                button=new Button[3];
                //chkbox=new Checkbox[3];
                scrbar=new Scrollbar[3];

                GridBagConstraints      c=new GridBagConstraints();
                GridBagLayout           g=new GridBagLayout();

                setSize(256,256);
                setLayout(g);

                for(int i=0;i<3;i++)
                {
                        Label   label=new Label("word"+i);
                        c.gridx=0;
                        c.gridy=i;
                        c.gridwidth=1;
                        c.weightx=0.1;
                        c.weighty=0.2;
                        g.setConstraints(label,c);
                        add(label);

                        text[i]=new TextField(16);
                        c.gridx=GridBagConstraints.RELATIVE;
                        c.gridwidth=2;
                        c.weightx=0.35;
                        g.setConstraints(text[i],c);
                        add(text[i]);

                        /*chkbox[i]=new Checkbox("not");
                        c.gridwidth=1;
                        c.weightx=0.15;
                        g.setConstraints(chkbox[i],c);
                        add(chkbox[i]);*/
```



```
                        scrbar[i]=new Scrollbar();
                        scrbar[i].setOrientation(Scrollbar.HORIZONTAL);
                        scrbar[i].setMaximum(100);
                        scrbar[i].setMinimum(0);
                        c.gridwidth=1;
                        c.weightx=0.3;
                        g.setConstraints(scrbar[i],c);
                        add(scrbar[i]);

                        button[i]=new Button(" select ");
                        c.gridx=GridBagConstraints.RELATIVE;
                        g.setConstraints(button[i],c);
                        add(button[i]);

                        button[i].addActionListener(this);
                }

                ok=new Button("   OK   ");
                c.gridx=1;
                c.gridy=3;
                c.gridwidth=1;
                c.weightx=0.7;
                c.weighty=0.2;
                g.setConstraints(ok,c);
                add(ok);

                cancel=new Button(" cancel ");
                c.gridx=GridBagConstraints.RELATIVE;
                c.gridy=3;
                c.weightx=0.3;
                g.setConstraints(cancel,c);
                add(cancel);

                ok.addActionListener(this);
                cancel.addActionListener(this);

                pack();
        }

        // Methods

        public String[] getWords()
        {
                String[]        words=new String[3];
                for(int i=0;i<3;i++) words[i]=text[i].getText();

                return words;
        }

        public double[] getValue()
        {
                double[]                val=new double[3];
                for(int i=0;i<3;i++)
                {
                        val[i]=scrbar[i].getValue()/90.0;
                        if(val[i]>1.0) val[i]=1.0;
                        System.out.println(scrbar[i].getValue());
                }

                return val;
        }

        public boolean input()
        {
                show();
```

```
                return isok;
        }

        public void actionPerformed(ActionEvent e)
        {
                if(e.getSource()==ok)
                {
                        isok=true;
                        dispose();
                }
                else if(e.getSource()==cancel)
                {
                        isok=false;
                        dispose();
                }

                for(int i=0;i<3;i++)
                {
                        if(e.getSource()==button[i])
                        {
                                WordListDialog  wd=new WordListDialog(parent,individu
al);
                                String          s;

                                wd.setLocation(getLocationOnScreen().x+64,getLocation
OnScreen().y+64);
                                s=wd.input();

                                if(s!=null) text[i].setText(s);
                        }
                }
        }
}
```

```
// WordListDialog.java (09/12/1997 Taku Yoshioka)

import java.awt.*;
import java.awt.event.*;
import java.util.*;

public class WordListDialog extends Dialog implements ActionListener
{
        // Fields

        private List            list;
        private Button          ok;
        private Button          cancel;
        private boolean         isok;
        private Frame           parent;

        // Constructors

        public WordListDialog(Frame p,Individual i)
        {
                super(p,"word",true);

                parent=p;
                isok=false;

                GridBagConstraints      c=new GridBagConstraints();
                GridBagLayout           g=new GridBagLayout();

                setLayout(g);

                list=new List();
                list.setMultipleMode(false);

                for(Enumeration e=i.words();e.hasMoreElements();)
                        list.add((String)(e.nextElement()));

                c.gridx=0;
                c.gridy=0;
                c.gridwidth=10;
                c.weightx=1.0;
                c.weighty=0.8;
                g.setConstraints(list,c);
                add(list);

                ok=new Button(" select ");
                c.gridy=1;
                c.gridwidth=1;
                c.weightx=0.7;
                c.weighty=0.2;
                g.setConstraints(ok,c);
                add(ok);

                cancel=new Button(" cancel ");
                c.gridx=GridBagConstraints.RELATIVE;
                c.weightx=0.3;
                g.setConstraints(cancel,c);
                add(cancel);

                ok.addActionListener(this);
                cancel.addActionListener(this);

                pack();
        }

        // Methods
```



```
        public void actionPerformed(ActionEvent e)
        {
                if(e.getSource()==ok)
                {
                        isok=true;
                        dispose();
                }
                else if(e.getSource()==cancel)
                {
                        isok=false;
                        dispose();
                }
        }

        public String input()
        {
                show();

                if(isok) return list.getSelectedItem();
                else return null;
        }
}
```

```
// ImageDialog.java (09/16/1997 Taku Yoshioka)

import java.awt.*;
import java.awt.event.*;

public class ImageDialog extends Dialog implements ActionListener
{
        // Fields

        private ImageCanvas     canvas;
        private Image           image;
        private Button          ok;

        // Constructors

        public ImageDialog(Frame parent,Image i)
        {
                super(parent,"Image",true);

                image=i;

                canvas=new ImageCanvas(image);
                canvas.setSize(256,256);
                add(canvas,"Center");

                ok=new Button("   OK   ");
                ok.addActionListener(this);
                add(ok,"South");

                pack();
        }

        // Events

        public void actionPerformed(ActionEvent e)
        {
                if(e.getSource()==ok) dispose();
        }
}
```

```
// TuboDialog.java (09/17/1997 Taku Yoshioka)

import java.awt.*;
import java.awt.event.*;
import java.util.*;

public class TuboDialog extends Dialog implements ActionListener
{
        // Fields

        private TextField       name;
        private Button          kansei;
        private Button          modify;
        private Button          ok;
        private double[]        evaluation;
        private Tubo            tubo;
        private Individual      individual;
        private Vector          evaldlgs;
        private Frame           parent;

        // Constructors

        public TuboDialog(Frame p,Tubo t,Individual i)
        {
                super(p,"Tubo Data",false);

                parent=p;
                tubo=t;
                individual=i;
                evaldlgs=new Vector();

                GridBagConstraints      c=new GridBagConstraints();
                GridBagLayout           g=new GridBagLayout();

                setLayout(g);

                Label   label=new Label("name");
                c.gridx=0;
                c.gridy=0;
                c.gridwidth=1;
                c.gridheight=1;
                c.weightx=0.1;
                c.weighty=0.1;
                g.setConstraints(label,c);
                add(label);

                name=new TextField(tubo.name(),16);
                c.gridx=GridBagConstraints.RELATIVE;
                c.gridwidth=7;
                c.weightx=0.7;
                g.setConstraints(name,c);
                add(name);

                ImageCanvas     canvas=new ImageCanvas(tubo.image());
                canvas.setSize(256,256);
                c.gridx=0;
                c.gridy=1;
                c.gridwidth=10;
                c.gridheight=10;
                c.weightx=0.8;
                c.weighty=0.8;
                g.setConstraints(canvas,c);
                add(canvas);

                ok=new Button("   OK   ");
                c.gridx=0;
```



```
                c.gridy=11;
                c.gridwidth=1;
                c.gridheight=1;
                c.weightx=0.33;
                c.weighty=0.1;
                g.setConstraints(ok,c);
                add(ok);

                kansei=new Button(" kansei ");
                c.gridx=4;
                g.setConstraints(kansei,c);
                add(kansei);

                modify=new Button(" modify ");
                c.gridx=7;
                g.setConstraints(modify,c);
                add(modify);

                ok.addActionListener(this);
                kansei.addActionListener(this);
                modify.addActionListener(this);

                pack();

                evaluation=i.output(tubo.feature());
        }

        // Events

        public void actionPerformed(ActionEvent e)
        {
                if(e.getSource()==ok)
                        dispose();

                else if(e.getSource()==kansei)
                {
                        try
                        {
                                WordListDialog  dlg1=new WordListDialog(parent,indivi
dual);
                                dlg1.setLocation(getLocationOnScreen().x+64,getLocati
onOnScreen().y+64);

                                String  word=dlg1.input();

                                if(word==null) return;

                                EvaluationDialog        dlg2=new EvaluationDialog(par
ent,tubo.name()+
                                                        ": "+word,1.0,evaluation[indi
vidual.word2Index(word)]);

                                dlg2.setLocation(getLocationOnScreen().x+64,getLocati
onOnScreen().y+64);
                                dlg2.show();
                                evaldlgs.addElement(dlg2);
                        }
                        catch(IllegalWordException ex) {ex.printStackTrace();}
                }

                else if(e.getSource()==modify)
                {
                        WordDialog      sd=new WordDialog(parent,individual);

                        sd.setLocation(getLocationOnScreen().x+64,getLocationOnScreen
().y+64);
```

```
                        if(sd.input()==true)
                        {
                                Hashtable       aux=new Hashtable();
                                String[]        word=sd.getWords();
                                double[]        val=sd.getValue();

                                for(int i=0;i<3;i++)
                                {
                                        if(word[i].compareTo("")==0) continue;
                                        aux.put(word[i],new Double(val[i]));
                                }

                                individual.modify(tubo.feature(),aux);
                                evaluation=individual.output(tubo.feature());
                        }
                }
        }

        public void dispose()
        {
                for(Enumeration e=evaldlgs.elements();e.hasMoreElements();)
                        ((Dialog)(e.nextElement())).dispose();

                super.dispose();
        }
}
```

```
3

wakawakasii
kihinngaaru
akarui

32

1.5     0.5     0.75    0.3333
wakawakasii     1.0
kihinngaaru     0.4
akarui          0.7
end

0.6666  0.25    1.0     0.33333
wakawakasii     0.55
kihinngaaru     0.1
akarui          0.4
end

1.5     0.25    0.3333  1.0
wakawakasii     0.5
kihinngaaru     0.9
akarui          0.6
end

1.0     2.0     0.75    0.3333
wakawakasii     0.1
kihinngaaru     0.3
akarui          0.45
end

0.6666  1.0     1.0     0.3333
wakawakasii     0.6
kihinngaaru     0.1
akarui          0.6
end

1.0     1.0     0.3333  0.1666
wakawakasii     0.3
kihinngaaru     0.6
akarui          0.4
end

0.6666  1.0     1.0     0.1666
wakawakasii     0.4
kihinngaaru     0.1
akarui          0.65
end

0.6666  2.0     0.75    0.3333
wakawakasii     0.25
kihinngaaru     0.55
akarui          0.5
end

1.0     1.0     0.3333  0.75
wakawakasii     0.6
kihinngaaru     0.6
akarui          0.2
end

1.5     4.0     0.3333  0.75
wakawakasii     0.7
kihinngaaru     0.6
akarui          0.6
```



```
end

0.6666  0.5     0.1666  0.3333
wakawakasii     0.4
kihinngaaru     0.55
akarui          0.3
end

1.5     2.0     0.1666  0.75
wakawakasii     0.6
kihinngaaru     0.6
akarui          0.6
end

0.75    0.25    0.1666  0.3333
wakawakasii     0.6
kihinngaaru     0.6
akarui          0.5
end

1.5     0.25    1.0     0.1666
wakawakasii     0.35
kihinngaaru     0.85
akarui          0.6
end

1.0     1.0     1.0     0.1666
wakawakasii     0.5
kihinngaaru     0.1
akarui          0.7
end

0.6666  0.5     0.75    0.3333
wakawakasii     0.5
kihinngaaru     0.45
akarui          0.35
end

1.5     4.0     1.0     0.1666
wakawakasii     0.6
kihinngaaru     0.25
akarui          0.85
end

1.5     1.0     0.75    0.3333
wakawakasii     0.9
kihinngaaru     0.55
akarui          0.5
end

1.0     4.0     0.3333  0.1666
wakawakasii     0.1
kihinngaaru     0.3
akarui          0.6
end

0.6666  4.0     1.0     0.3333
wakawakasii     0.6
kihinngaaru     0.1
akarui          0.35
end

1.0     2.0     0.1666  0.3333
wakawakasii     0.4
kihinngaaru     0.35
akarui          0.5
```

```
end

0.3333  1.0     0.75    0.3333
wakawakasii     0.5
kihinngaaru     0.35
akarui          0.3
end

1.0     0.25    0.1666  1.0
wakawakasii     0.5
kihinngaaru     0.45
akarui          0.4
end

1.5     0.5     0.3333  0.1666
wakawakasii     0.6
kihinngaaru     1.0
akarui          0.45
end

0.6666  4.0     0.1666  0.3333
wakawakasii     0.05
kihinngaaru     0.6
akarui          0.6
end

1.0     2.0     0.3333  0.75
wakawakasii     0.6
kihinngaaru     0.3
akarui          0.35
end

1.5     4.0     0.1666  0.75
wakawakasii     0.65
kihinngaaru     0.55
akarui          0.75
end

1.5     0.5     0.1666  0.1666
wakawakasii     0.4
kihinngaaru     0.6
akarui          0.0
end

0.6666  2.0     0.3333  0.3333
wakawakasii     0.3
kihinngaaru     0.3
akarui          0.35
end

1.0     0.25    0.75    0.1666
wakawakasii     0.55
kihinngaaru     1.0
akarui          0.35
end

1.5     1.0     1.0     0.3333
wakawakasii     0.8
kihinngaaru     0.5
akarui          0.65
end

1.0     4.0     0.3333  0.75
wakawakasii     0.7
kihinngaaru     0.0
akarui          0.4
```



```
end
```

```
32

tubo1
"/home/yoshioka/java/kansei/data/6*4-1:2-1:6:1-1:1:1.1.pic.gif"
1.5     0.5     0.75    0.3333

tubo2
"/home/yoshioka/java/kansei/data/4*6-1:4-1-1:1:1.1.pic.gif"
0.6666  0.25    1.0     0.33333

tubo3
"/home/yoshioka/java/kansei/data/6*4-1:4-1:1:1-1.1.pic.gif"
1.5     0.25    0.3333  1.0

tubo4
"/home/yoshioka/java/kansei/data/6*6-2:1-1:6:1-1:1:1.1.pic.gif"
1.0     2.0     0.75    0.3333

tubo5
"/home/yoshioka/java/kansei/data/4*6-1:1-1-1:1:1.1.pic.gif"
0.6666  1.0     1.0     0.3333

tubo6
"/home/yoshioka/java/kansei/data/6*6-1:1-1:1:1-5:2:5.1.pic.gif"
1.0     1.0     0.3333  0.1666

tubo7
"/home/yoshioka/java/kansei/data/4*6-1:1-1-5:2:5.1.pic.gif"
0.6666  1.0     1.0     0.1666

tubo8
"/home/yoshioka/java/kansei/data/4*6-2:1-1:6:1-1:1:1.1.pic.gif"
0.6666  2.0     0.75    0.3333

tubo9
"/home/yoshioka/java/kansei/data/6*6-1:1-1:1:1-1:6:1.1.pic.gif"
1.0     1.0     0.3333  0.75

tubo10
"/home/yoshioka/java/kansei/data/6*4-4:1-1:1:1-1:6:1.1.pic.gif"
1.5     4.0     0.3333  0.75

tubo11
"/home/yoshioka/java/kansei/data/4*6-1:2-5:2:5-1:1:1.1.pic.gif"
0.6666  0.5     0.1666  0.3333

tubo13
"/home/yoshioka/java/kansei/data/6*4-2:1-5:2:5-1:6:1.1.pic.gif"
1.5     2.0     0.1666  0.75

tubo14
"/home/yoshioka/java/kansei/data/4*6-1:4-5:2:5-1:1:1.1.pic.gif"
0.75    0.25    0.1666  0.3333

tubo15
"/home/yoshioka/java/kansei/data/6*4-1:4-1-5:2:5.1.pic.gif"
1.5     0.25    1.0     0.1666

tubo17
"/home/yoshioka/java/kansei/data/6*6-1:1-1-5:2:5.1.pic.gif"
1.0     1.0     1.0     0.1666

tubo18
"/home/yoshioka/java/kansei/data/4*6-1:2-1:6:1-1:1:1.1.pic.gif"
0.6666  0.5     0.75    0.3333
```



```
tubo19
"/home/yoshioka/java/kansei/data/6*4-4:1-1-5:2:5.1.pic.gif"
1.5     4.0     1.0     0.1666

tubo21
"/home/yoshioka/java/kansei/data/6*4-1:1-1:6:1-1:1:1.1.pic.gif"
1.5     1.0     0.75    0.3333

tubo22
"/home/yoshioka/java/kansei/data/6*6-4:1-1:1:1-5:2:5.1.pic.gif"
1.0     4.0     0.3333  0.1666

tubo24
"/home/yoshioka/java/kansei/data/4*6-4:1-1-1:1:1.1.pic.gif"
0.6666  4.0     1.0     0.3333

tubo25
"/home/yoshioka/java/kansei/data/6*6-2:1-5:2:5-1:1:1.1.pic.gif"
1.0     2.0     0.1666  0.3333

tubo26
"/home/yoshioka/java/kansei/data/4*6-1:1-1:6:1-1:1:1.1.pic.gif"
0.3333  1.0     0.75    0.3333

tubo30
"/home/yoshioka/java/kansei/data/6*6-1:4-5:2:5-1.1.pic.gif"
1.0     0.25    0.1666  1.0

tubo31
"/home/yoshioka/java/kansei/data/6*4-1:2-1:1:1-5:2:5.1.pic.gif"
1.5     0.5     0.3333  0.1666

tubo32
"/home/yoshioka/java/kansei/data/4*6-4:1-5:2:5-1:1:1.1.pic.gif"
0.6666  4.0     0.1666  0.3333

tubo33
"/home/yoshioka/java/kansei/data/6*6-2:1-1:1:1-1:6:1.1.pic.gif"
1.0     2.0     0.3333  0.75

tubo34
"/home/yoshioka/java/kansei/data/6*4-4:1-5:2:5-1:6:1.1.pic.gif"
1.5     4.0     0.1666  0.75

tubo36
"/home/yoshioka/java/kansei/data/6*4-1:2-5:2:5-5:2:5.1.pic.gif"
1.5     0.5     0.1666  0.1666

tubo37
"/home/yoshioka/java/kansei/data/4*6-2:1-1:1:1-1:1:1.1.pic.gif"
0.6666  2.0     0.3333  0.3333

tubo38
"/home/yoshioka/java/kansei/data/6*6-1:4-1:6:1-5:2:5.1.pic.gif"
1.0     0.25    0.75    0.1666

tubo39
"/home/yoshioka/java/kansei/data/6*4-1:1-1-1:1:1.1.pic.gif"
1.5     1.0     1.0     0.3333

tubo40
"/home/yoshioka/java/kansei/data/6*6-4:1-1:1:1-1:6:1.1.pic.gif"
1.0     4.0     0.3333  0.75
```

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2//EN">
<!--NewPage-->
<html>
<head>
<!-- Generated by javadoc on Thu Sep 25 17:45:55 GMT+09:00 1997 -->
<title>
  Class Hierarchy
</title>
</head>
<body>
<a name="_top_"></a>
<pre><a href="packages.html">All Packages</a>  <a href="AllNames.html">Index</a></pre
><hr>
<h1>
  Class Hierarchy
</h1>
<ul>
  <li> class java.lang.Object
  <ul>
    <li> class java.awt.Component (implements java.awt.image.ImageObserver, java.awt.
MenuContainer, java.io.Serializable)
    <ul>
      <li> class java.awt.Canvas
      <ul>
        <li> class <a href="BarCanvas.html#_top_">BarCanvas</a>
        <li> class <a href="ImageCanvas.html#_top_">ImageCanvas</a>
      </ul>
      <li> class java.awt.Container
      <ul>
        <li> class java.awt.Window
        <ul>
          <li> class java.awt.Dialog
          <ul>
            <li> class <a href="DataListDialog.html#_top_">DataListDialog</a> (implem
ents java.awt.event.ActionListener)
            <li> class <a href="EvaluationDialog.html#_top_">EvaluationDialog</a> (im
plements java.awt.event.ActionListener)
            <li> class <a href="ImageDialog.html#_top_">ImageDialog</a> (implements j
ava.awt.event.ActionListener)
            <li> class <a href="NewDataDialog.html#_top_">NewDataDialog</a> (implemen
ts java.awt.event.ActionListener)
            <li> class <a href="TuboDialog.html#_top_">TuboDialog</a> (implements jav
a.awt.event.ActionListener)
            <li> class <a href="WordDialog.html#_top_">WordDialog</a> (implements jav
a.awt.event.ActionListener)
            <li> class <a href="WordListDialog.html#_top_">WordListDialog</a> (implem
ents java.awt.event.ActionListener)
          </ul>
          <li> class java.awt.Frame (implements java.awt.MenuContainer)
          <ul>
            <li> class <a href="MainWindow.html#_top_">MainWindow</a> (implements jav
a.awt.event.ActionListener)
          </ul>
        </ul>
      </ul>
    </ul>
    <li> class <a href="Individual.html#_top_">Individual</a> (implements java.io.Ser
ializable)
    <li> class java.lang.Throwable (implements java.io.Serializable)
    <ul>
      <li> class java.lang.Exception
      <ul>
        <li> class <a href="IllegalWordException.html#_top_">IllegalWordException</a>
      </ul>
    </ul>
    <li> class <a href="Tubo.html#_top_">Tubo</a> (implements java.io.Serializable)
```



```
  </ul>
</ul>
</body>
</html>
```