

統計パッケージのグラフィカル ユーザインタフェース

——SHAZAM, Tcl/Tk, WWW を利用して——

中 野 純 司

1 はじめに

計算機は当初、その名称どおり大量の数値計算を実行するために開発された。したがって、扱わなければならないデータ量が多く、複雑な計算を繰り返し行う必要のある統計計算プログラムが初期のうちから重要な計算機ソフトウェアであったことは当然と言えよう。多くの科学技術分野の専門家が統計パッケージを利用して研究を行ったのである。

その発祥以来、計算機技術は着実に発展を遂げてきたが、パーソナルコンピュータの出現は計算機の質的転換をうながすことになったという点で特筆すべきであろう。計算機室で専門家のための数値計算を行う機械から、あらゆる場所で万人のために多様な情報処理を行うための機械になる可能性が開けたのである。

そして、インターネット技術、特に WWW (World Wide Web) 技術は文字情報だけではなく、音声、静止画、動画などのマルチメディア情報を総合的に容易に、しかもどこでも扱うことを可能にした。このことは情報処理端末としての計算機の爆発的な普及を引き起こした。

統計計算プログラムもこのような動きとは無関係ではいられない。現在では統計パッケージとしてまとめられている統計計算プログラムの目的は、記述統計量の計算やモデルに関する推論をできるだけ容易に行うことである。また、統計解析は探索的なものであるでいろいろな手法を柔軟に適用できなければならない。これまでの努力により、手法の種類やその計算の信頼性に関しては現在ではほぼ満足できる状況になっている。まだ改良が必要な点は、それらを易しく、できるだけ間違わないように利用できるように使い勝手を良くすることである。それは統計パッケージが統計の専門家だけではなく、統計解析を行う必要があるが必ずしも統計学に詳しくない、いろいろな分野の専門家にも、パーソナルコンピュータの普及により容易に利用されるようになってきたことから重要である。

統計パッケージを使い易くするための研究としては、人工知能技術を利用して統計手法の誤用を防ごうとするもの、計算機上での統計モデルの表現、直感的なグラフィカルユーザインタフェース (Graphical User Interface, GUI), などがある。この中でも GUI は統計パッケージの入口とも言うべきものであり、その使い易さが初心者にとっては決定的な影響を与える。

そこで本稿では、現時点における望ましい統計パッケージの GUI を考察する。まず、これまでの統計パッケージの資産と現在利用できる技術を効果的に組み合わせて GUI を作成する方法を考察する。そして統計パッケージ SHAZAM に対して GUI を実装する。さらに GUI の標準となりつつある WWW ブラウザ上で同様の GUI を利用することを提案する。

次節では統計パッケージの発展とそのユーザインタフェースの基本的な考え方を述べる。3 節では本稿で用いる統計パッケージである SHAZAM を概説する。4 節では現時点で統計パッケージのユーザインタフェースを考えるとときに利用できる技術について述べる。5 節では Tcl/Tk により試

作した SHAZAM の GUI について述べる。6 節では 5 節で作成したものをもとにして WWW ブラウザを利用した GUI を解説する。最後の 7 節では今後の統計パッケージの GUI について考察する。

2 統計パッケージとそのユーザインタフェース

2.1 統計パッケージの発達

初期の統計ソフトウェアは、主として数値計算のために開発された最初の高級計算機言語である Fortran 言語のサブルーチン集であった。よく使われる複雑な手順の統計計算をライブラリ化することにより、データ処理プログラムを書くことを容易にしたのである。データの変換やいくつかの統計解析を連続して行うためには、Fortran 言語でその手続きを記述しなければならなかった。このようなライブラリは各社の大型計算機の附属ソフトウェアとして提供されることが多かった。

ところが、Fortran 言語は汎用言語であるために、統計解析でよく利用するような手続きが簡単に書けるとは言いにくい。そこで、統計解析に特化してそれらをわかりやすく記述できるようなシステムまたは言語が、応用分野の統計利用者から要求されるようになってきた。それに答えて、標準的な統計解析手法とデータ管理および変換機能をそなえ、習得の容易な言語でそれらを利用できる“統計パッケージ”が開発されたのである。BMDP, SPSS, SAS などがこのようにしてできた有名な統計パッケージである。

統計パッケージを利用するには、それに特有の言語でプログラムを書き、それが完成した時点で統計パッケージにそれを読み込ませてまとめて実行するというバッチ処理を行う。統計パッケージは統計解析用言語のインタプリタであるから、最初にそれを起動しておき、そのプロンプトに対して

実行したい命令を与えるという対話的な使い方ができるものも多くなった。そのような使い方では統計パッケージは文字を介して利用者と情報のやり取りを行うことになるが、このユーザインタフェースをCUI (Character User Interface) という。このような統計パッケージは応用分野でデータ解析を行う者にとっては欠くことのできない道具となった。一部の統計専門家からは、統計パッケージのせいで手法の十分な理解をしないまま単にデータを計算機に入力するだけの形式的な統計解析が多くなった、と言う“統計パッケージ有害論”が出されることもあった。しかし大型計算機を利用する（できる）人は限られた専門家だけであったので、この問題はそれほど影響のあるものではなかった。

パーソナルコンピュータの出現は統計パッケージにも大きな影響を与えた。大型計算機上の多くの統計パッケージはパーソナルコンピュータに移植され、容易に利用できるようになった。それらはパーソナルコンピュータのマーケットの拡大とともに低価格になり、それがまた利用者数の増大を招いた。統計学の知識がほとんどない人でも統計パッケージを利用できるようになってきたのである。そこで、利用者の大多数を占めるようになった初心者のための配慮が統計パッケージに求められるようになった。

さて、データの要約のためにグラフが有力な手段であることは明らかであったが、大型計算機では文字を利用した貧弱なグラフしか一般には描くことができなかった。ところが、パーソナルコンピュータはその初期の頃からビットマップディスプレイを備えていた。したがって美しいグラフィックスを描くことが非常に容易であった。さらに、対話の方法として画面上にメニューの一覧を表示し、その一つを指定するとさらに詳しいメニューを出現させるようなユーザインタフェースも利用できるようになった。自分で命令を打ち込むのではなく、与えられたメニューから必要な命令を選択することにより統計解析ができるわけである。このようなグラフィッ

クスを多用した計算機との対話法を GUI (Graphical User Interface) と呼ぶ。大型計算機からの移植とは別に、最初からパーソナルコンピュータ用に作成された統計パッケージも出現したが、それらは初歩的な GUI を備えているのが普通であった。そのような統計パッケージには SYSTAT, STATGRAPHICS などがある。

ところで、計算機ハードウェアと利用者を結ぶ基本ソフトウェアであるオペレーティングシステム (Operating System, OS) は CUI が基本であった。大型計算機の OS は (現在でも) ほとんど CUI であり、ワークステーション上の標準 OS である UNIX も、それを模範としたパーソナルコンピュータの OS である MS-DOS も CUI であった。SYSTAT や STATGRAPHICS は MS-DOS 上に独自の GUI を作成したものであったが、その GUI としての完成度はあまり高くない。一般に GUI は CUI に較べて、はるかに直感的で使い易いので、その必要度は初心者ほど切実なものがある。コンピュータが急速に普及するにつれ、OS 自身に GUI が取り込まれるのは当然のことであった。Macintosh パーソナルコンピュータの OS は最初から GUI をそなえていた。ワークステーションでは X Window System が標準的な GUI を提供するようになった。そして、MS-DOS も Microsoft Windows という GUI を持つにいたった。

一般的に言って、GUI のプログラミングは CUI のプログラミングに較べてはるかに難しかった。その使い易さとは逆に、Macintosh 上のアプリケーション開発が難しいことは有名であった。X Window System や Microsoft Windows 上でただ一行 “Hello, world!” と表示するプログラムが、当初、数十行を要したこともある。その上、GUI プログラムはプラットフォーム間での移植が面倒である。CUI を持つプログラムであれば、そのソースコードにわずかな変更を施しコンパイルし直せば、ただちに異なる OS 上で利用できることが多い。それに較べて、X Window

System のプログラムを Microsoft Windows のプログラムとして書き直すにはインタフェース部分のほとんどすべての変更が必要となる。また GUI に関する部分は、プログラムの本質的な部分（統計パッケージで言えば統計計算のためのプログラム）より多くのプログラムが必要であることが一般的である。MS-DOS 上でもっとも成功した統計パッケージであると言われた SYSTAT が、1994 年に SPSS に吸収された原因のひとつが Microsoft Windows への対応の遅れであったと言われていることは興味深い。

統計パッケージで大切なことは、必要な手法が完備しており、その計算が常に正確に行われることである。数値計算の誤差については注意深くプログラムされていなければならないし、きわめて大量のデータ処理を行っても計算が破綻してはいけない。このようなパッケージの手法数と信頼性に関して、歴史のある統計パッケージはほとんど満足できるものになっていた。それに対して、パーソナルコンピュータ用に開発されたものは、当初問題のあるものもあった。しかし、よく利用されたものは、その利用者の数が多かったこともあり、急速に改善された。現在のほとんどの商用統計パッケージには、充分な手法と信頼性があると考えてよい。

したがって、次の段階として“使い易さ”という要素が重要になってくる。これは初心者利用が増えたことから大切なことである。現在、使いやすいシステムを作成するために、いくつかの方向で研究が進んでいる。

ひとつは柔軟性である。これは特に統計学の専門家からの要求が強い。パッケージ開発者の提供する手法だけでは、それがいかに多様であろうともすべての利用者を満足させることはありえない。利用者自身が新しい手法を開発でき、その際、複雑な構造のモデルをできるだけ容易に、またその推測手順のアルゴリズムを直接的に記述できるようなくみが必要とされる。また、開発したものが全体のシステムの中に自然に整合するもので

なければならない。このような柔軟な能力を持つシステムの例としては S 言語 (Chambers and Hastie, 1992) がある。

もうひとつはシステムの知能化である。統計パッケージに主として含まれているのは統計計算手法である。現在の統計パッケージにおいては、これまでに提案されている手法はほとんどすべてが、組み込まれているといっってよい。組み込まれている手法が多すぎて、専門の統計学者でさえ、そのすべてを知っているわけではないほどである。このような状態であるので、専門家でない利用者が適用すべき手法がわからなかったり、迷ったりすることは当然である。そこで、これらの点をパッケージ自身にアドバイスさせるような機能を持たせることが要求されるようになっている。それを実現するためには、知識工学の手法を利用することが考えられる。知識工学の手法はいわゆるエキスパートシステムの作成を行うために研究されたものである。ここでは主としてパターンマッチングによる推論が行われ、いろいろな場合分けによる推論を効果的に選択する。この方向においては、いくつかの研究が行われてきた（例えば、中野、山本、岡田 1991）が、人間の知的活動を模倣しようとする他の研究と同様、難しい問題が山積しており、まだまだ満足のいく結果にはいたっていないと言ってよい。

使いやすさを追求するためのもうひとつの研究が GUI の利用である。以前と異なり、現在では GUI 開発のためのツールがかなり充実している。われわれが利用する Tcl/Tk 言語ならば “Hello World!” と書いたウィンドウを出すためのプログラムは 2 行で記述できる。また、OS 間の移植も容易になっている。さらに、WWW ブラウザのような GUI の共通プラットフォームも普及してきた。本稿では現時点で利用できるこれらの技術を利用して、できるだけ簡単に有用な統計パッケージの GUI を作成することを考える。

2.2 統計パッケージのユーザインタフェース

現在の統計パッケージではGUIが使えるものが多いが、CUIも滅びたわけではない。それはGUIとCUIには、それぞれ短所と長所があるからである。このことを明確に指摘したのはLiu, Chan, Montgomery and Muller (1995)である。

CUIは、プログラムが利用者に入力をうながし、利用者は文字列の形式で入力を与え、プログラムはそれを読んで処理を行う、という単純なプログラム操作法である。そのもっとも大きな短所は命令を利用者が記憶しなければならないことである。ファイルを消去するだけでも、erase, delete, del, remove, rmなど考えられるもののなかで、どれを用いるかを知っていなければならない。したがって、初心者やたまにしかその統計パッケージを利用しない人にとっては言語を覚えなければならないことが非常な負担となる。そのかわり、一度それをマスターしてしまえばパッケージの機能を直接指定できるわけで処理は速いし、プログラムを書くことにより複雑な操作も指定できる。

GUIでは、ウィンドウ、メニュー、ダイアログボックス、アイコンなどのオブジェクトをマウスなどで指定することによりプログラムの操作を行う。したがって、一般にプログラム処理は決められた順序で行われるのではなく、マウスのクリックなどの利用者が与えた指令に応じて処理を行うイベント駆動型となる。GUIでは画面に表示される情報量が多く、ボタンの上に表示されている文字、プルダウンメニューを利用すれば、例えばファイルを消去することなどは容易に行える。また、OSで操作法がある程度統一されるので、それを覚える機会が多い。また、テキストとグラフィックスを同時に扱うことができる。ただ、GUIにも欠点はある。GUIだけでは使用履歴が保存できないし、いくつかの操作をひとつにまとめることも難しい。また入力と出力を同等に扱い、いくつかの機能を逐

次的に組み合わせるようなことは困難である。

これらの考察より、Liu, Chan, Montgomery and Muller (1995) は CUI と GUI を同時に用いる混合ユーザインタフェース (Composite user interface) を提案した。これは CUI を持つシステムにその CUI を利用するための GUI プログラムを付加し、それらが互いにやりとりを行って作業を進めるというものである。利用者は普通は GUI だけを利用すればよいが、必要があれば CUI を利用することもできる。かれらはこの方針にしたがって、CUI を持つ SCA 統計システムに対して、GUI 用のフロントエンドとして SCAWIN というプログラムを作成した。

現在の計算機環境はかれらの時点に較べても大きく変化している。特に GUI 作成のためのツールの充実と、インターネットの普及が重要である。本稿ではこれらの技術を取り入れて、統計パッケージ SHAZAM に対して混合インタフェースを作成することを考える。

3 統計パッケージ SHAZAM

本稿においては統計パッケージの具体例として SHAZAM を取り上げ、そのための GUI を作成する。SHAZAM は Canada の University of British Columbia に所属する Kenneth J. White が中心となって 1970 年代後半から開発、販売している主として計量経済学のために作られた統計パッケージである。現在、Version 8.0 が利用されている (White 1997)。プログラムは (ほとんど) すべて Fortran 言語で書かれており、ソースコードは約 55000 行にのぼる。

3.1 SHAZAM の機能

SHAZAM の特徴としては：

- 計量経済学の手法が豊富である。
- 価格が安い。またサイトライセンスをとれば構成員には無料でコピーを配布できる。
- 多種のマシンで稼動する。すなわち、MS-DOS, Windows 3.1, Windows 95, Windows NT, Macintosh, 種々の UNIX, 大型汎用機で動く。
- インターネットに積極的に対応している。例えばバージョンアップなどは直接ダウンロードすることができる。

などがある。

利用できる手法（とそのためのコマンド）をあげてみると：

- 入出力
ファイル指定 (FILE), 読む (READ), 書く (WRITE, PRINT),
書式指定 (FORMAT)
- 変数の取り扱い
範囲指定 (SAMPLE, TIME), 変数の生成など (GENR, GEN1, IF,
IF1, SKIPIF, DELETE, NAMES, RENAME)
- 基礎統計学
平均・分散・共分散など (STAT), 種々の確率分布 (DISTRIB)
- グラフ
ヒストグラム・散布図 (PLOT)
- 重回帰分析
線形回帰・ステップワイズ回帰・分布ラグモデル・正確な Durbin-Watson 検定・Ridge 回帰 (OLS), 回帰係数の線形仮説の検定 (RESTRICT, TEST), 回帰係数の信頼区間および信頼楕円 (CONFID), 回帰係数の線形および不等式制約 (RESTRICT, BAYES), 不等分散モデル (GLS), 残差が AR または ARMA の場合 (AUTO), 残差

が非正規の場合 (MLE), 回帰診断 (DIAGNOS), 予測 (FC)

- さらに高度な回帰分析

Box-Cox 回帰・Box-Tidwell 回帰 (BOX, LAMBDA), ロバスト回帰 (ROBUST), 非線形回帰 (NL, EQ, COEF, END), Probit 回帰 (PROBIT), Logit 回帰 (LOGIT), Tobit 回帰 (TOBIT), クロスセクションで時系列の場合 (POOL), 時変化係数回帰 (FLS), 一般エントロピー法 (GME)

- 時系列解析

1 変量 ARIMA モデルの同定・推定・診断・予測 (ARIMA), 共和分と単位根の検定 (COINT), ARCH, GARCH モデル (HET)

- 計量経済学

2 段階最小 2 乗法 (2SLS), 同時方程式モデル (SYSTEM), 指数 (INDEX)

- 行列計算 (MATRIX, DIM, COPY)

- その他の統計・数学手法

主成分分析 (PC), ノンパラメトリック密度および回帰推定 (NONPAR), 線形計画法 (LP), 並べ変え (SORT), 数値積分 (INTEG), 数値微分 (DERIV)

- プログラミング (DO, ENDO, ENDIF, PROC, PROCEND, EXEC)

- システム管理 (STOP, PAR, SIZE, COMPRESS, SET, DISPLAY, CHECKOUT, PAUSE, SAVE, RESTORE, TITLE, DUMP)

- その他 (HELP, DEMO, MENU)

となる。

SHAZAM のひとつの命令は一行で構成され,

コマンド 引数 1 … 引数 m / オプション 1 … オプション n

のような形式をとる。引数やオプションはないこともある。長い命令の場

合は行の最後に&を付加しておけば次行も同一行とみなす継続行となる。特に指定しない限り SHAZAM は英大文字と英小文字を区別しない（すべて英大文字として扱う）。

ここで SHAZAM のプログラム例として簡単なものをあげ、使用されているいくつかのコマンドを説明する。なお、行頭の括弧部は説明のために行番号をつけたものでプログラムの一部ではない。

- (1) sample 1 43
- (2) read (rent.dat) no rent pos dist area age grade
- (3) print rent area dist age / beg=10 end=20
- (4) skipif (rent .gt. 10.0)
- (5) ols rent area dist age / influence &
- (6) predict=pred resid=res
- (7) plot res pred
- (8) stop

(1) sample コマンド

使用するデータ範囲の始めと終りを指定する。ここでは読み込むデータの長さが 43 であることを指示する。

(2) read コマンド

データをファイルから変数に読み込む。データを保存してあるファイル rent.dat には no から grade までの 7 つの変数に対応する 43 個の数値が列ごとにテキスト形式で書き込まれている。

(3) print コマンド

変数をプリントする。出力する変数名を引数として並べるとともに、オプションでプリントすべき期間を観測値番号を用いて指示した。

(4) skipif コマンド

条件にあうものは計算の際に無視する。ここでは変数 rent の値が

10.0 より大きい観測値は次行以下の計算に用いない。

(5)(6) `ols` コマンド

最小 2 乗法 (Ordinary Least Squares method) の計算を行う。第一引数 (con) が従属変数であり、第二引数 (以降) が独立変数である。特に指定しなければ定数項が独立変数として含まれる。オプションは回帰診断のためのいくつかの統計量を印刷し (influence)、予測値を変数 `pred` に格納し、残差を変数 `res` に格納することを指定する。なお、(5) 行は継続行である。

(7) `plot` コマンド

散布図を描く。最後の (ここでは第二) 引数で与えられた変数が横軸に取られ、第一引数から最後から 2 番目までの引数で与えられた変数が縦軸に取られる。ここでの散布図は文字を利用した CUI 的なものだが、オプション `GNU` を用いるとよく利用されているフリーのグラフ作成ソフトウェア `GNUPLOT` を利用して、高精度なグラフを描くこともできる。

(8) `stop` コマンド

`SHAZAM` を終了する。

3.2 SHAZAM のユーザインタフェース

`SHAZAM` は非常に古典的な統計パッケージである。そのユーザインタフェースは `SHAZAM` 言語を用いてプログラムを書き、それを `SHAZAM` プログラムに渡して計算を行うというものである。

もっとも簡単な使いかたとしては `SHAZAM` を起動し、プログラムの各行をキーボードから一行ずつ入力すればよい。このような対話的な使い方が便利なコマンドとして `DEMO`, `HELP` コマンドがある。`DEMO` は `SHAZAM` のごく基本的な命令を紹介した対話的な入門解説であり、ひと

まとまりの解説のあとで利用者が自分で命令を打ち込んでそれらを確認する、という形式をとる。また、HELP はすべての命令の簡単な解説を、対話処理の中で見ることができるというものである。DEMO は初心者が最初に試せば SHAZAM の使い方を習得するのに非常に有効であり、HELP も対話的に利用している場合には役にたつ。

しかし、SHAZAM を対話的に使うことはあまりすすめられたものではない。それは、入力した行の修正がしにくい、過去の入力の再利用が困難である、など不便な点が多いからである。したがって普通はエディタを用いて、先の例のようなプログラムをファイルに格納し（ファイル名を例えば example1.sha とする）、それをバッチ的に実行する。ひとつの方法として SHAZAM のプロンプトに対して file input example1.sha とすると、そのファイルの内容があたかもキーボードから入力されたかのように計算を始める。また、最初に file screen example1.out とするとそれ以後の作業の結果は画面に出ると同時に指定したファイル example1.out にも書き出される。

UNIX OS の場合、シェルのプロンプト (%) に対して

```
% shazam < example1.sha > example1.out
```

としても若干の出力形式の違いを除いて、ほとんど同じ結果がえられる。これは UNIX の入出力の切替え機能を利用したもので、SHAZAM を起動し、入力を example1.sha から行い、出力を example1.out に行くことを指示する。このようにバッチ的に処理するときには、プログラムファイル (example1.sha) の最後の行を stop としておくと、処理のあと SHAZAM が終了し、UNIX のプロンプトが出る。

なお、SHAZAM の Microsoft Windows 版および Macintosh 版にはエディタウィンドウと SHAZAM ウィンドウを組み合わせた簡単な GUI があるが、新しい機能として付加されたものはほとんどない。

4 GUI 作成のための技術

4.1 Tcl/Tk

2節で述べたように GUI プログラムを作成することは当初非常に難しかった。ところが現在のようにほとんどすべての OS が GUI を利用するようになると、どのようなプログラムを書くにしても GUI プログラムを避けて通ることができなくなった。

そこでまず、GUI プログラムのためのライブラリが開発された。これは OS（または OS に標準的な GUI システム）がサポートする機能を使いやすいようにまとめた、言語ごとのサブルーチン集である。このようなライブラリは OS とプログラム言語に完全に依存する。これにより GUI 作成の手間はかなり軽減されたが、それでもまだ GUI プログラムの専門家のためのものといわざるを得なかった。計算機を利用している統計学の専門家が気楽に利用するには荷が重かったのである。

さらに使い易いものを、という要求に応じて、言語の基本機能として GUI をサポートするものがあらわれた。その最初に普及したものが Tcl/Tk (Ousterhout 1994) であった。

Tcl (Tool Command Language) は当時 University of California at Berkeley に所属していた John K. Ousterhout によって作成された。これは多くの応用プログラムを開発し、それぞれにおいて異なった組み込み言語を作成した経験から、共通に使える組み込み言語として Tcl を設計したのである。

Tcl は文字列とリストを基本構造としたインタプリタ言語で、“括弧のない Lisp”と言われることもある。プログラムの読みやすさにも配慮されており、専門家でなくても使いやすい。リスト処理および文字列の処理

が強力で、既存のソフトウェアを組み合わせるのに有用である。

Tk (Tool Kit) は Tcl に GUI 作成機能を付加するものである。Tk にはウィジェットと呼ばれる GUI 用の基本的な部品 (ラベル, ボタン, ラジオボタン, スケール, メニュー, テキスト, キャンバスなど) が完備している。それらにマウスカーソルをあわせる, 右ボタンを押す, などのイベント発生時の動作を定義することができ, イベント駆動のプログラムが容易に書ける。また, 多くのウィジェットを画面上に簡単に配置する仕組みがある。例えば先に述べた画面上に “Hello, world !” と表示するプログラムは Tcl/Tk では

```
button .b -text "Hello, world !" -command exit
pack .b
```

のように簡単に書ける。

Tcl/Tk はその使い易さのために急速に普及した。また作者が処理系のソースプログラムを無料で配布し, その利用に関してもほとんど条件をつけなかったことも普及した理由のひとつだろう。ただ, 作者の当初の意図とは違って, Tcl/Tk が他のシステムの組み込み言語として利用されることは少なく, Tcl インタプリタである tclsh や Tcl/Tk インタプリタである wish がそのまま利用されることが多く, 一種のスクリプト言語となった。統計パッケージの分野でも, 山本・中野・田村 (1995) が wish を利用した。

Ousterhout はその後, Sun Microsystems 社に移り, 精力的に Tcl/Tk のバージョンアップを続けている。そして最近, Tcl/Tk をインターネットの WWW 上の共通スクリプト言語 (Tclet) にすることを試みている。これについては別項で述べる。

4.2 インターネットと WWW

インターネットは地球規模のコンピュータネットワークである。現時点において、これに対応しうまく利用することがもっとも重要な計算機利用法であるとみなされている。この動きは計算機の歴史においてパーソナルコンピュータの出現に次ぐ大変革であると言えるだろう。

ローカルエリアネットワークを結合するというインターネットの考え方はかなり古いものである。1960年代の米ソ冷戦時代、アメリカ軍部は、核戦争にも耐えられるような情報処理システムがほしかった。そのためには集中処理システムは不適切であり、一部が破壊されても全体としての機能がとまってしまうことのないような分散処理システムが必要であった。また、計算機研究者もアメリカ全土に分散しており、お互いの研究成果を連絡しあうためにもコンピュータネットワークの必要性が高まっていた。それらに答えてできた初めての広域ネットワークが ARPA ネットである。その後技術的な進歩と規格の統一が進み、1980年代にはほぼ現在のようなインターネットの仕組みができあがっていた。

インターネットが現在のように爆発的に普及するきっかけをつくったのは WWW (World Wide Web) の発表である。WWW はスイスのヨーロッパ高エネルギー研究所で所員の連絡のための電子掲示板のために作られた。情報伝達のための通信用プロトコル HTTP (HyperText Transfer Protocol) と掲示内容のフォーマットを規定する HTML (HyperText Markup Language) からなる。HTML で書かれた文書のなかにはイメージ、音声、動画などのファイルを指定して、文書の一部に埋め込むことができる。さらに、他の計算機上にある HTML ファイルを指定することもできる。これらのファイルを閲覧するにはブラウザと呼ばれるプログラムを利用する。ブラウザプログラムの例としては、Netscape Navigator や Microsoft Internet Explorer などがある。

ブラウザプログラムは閲覧したいファイルのあるサーバ計算機上の HTTP daemon プログラムにファイル送信の要求を送る。HTTP daemon プログラムはその要求に答えてファイルを送信したあと、ただちに接続を終了する。ファイルを受け取ったブラウザプログラムは HTML で書かれている内容を解釈し、画面上に表示する。

このような WWW が爆発的に普及した理由のひとつは、その使いやすくなりやすいユーザインタフェースにある。最近のブラウザプログラムでは、基本的な操作はすべてマウスやトラックボールのようなポインティングデバイスで画面上のオブジェクトをクリックするだけである。したがって、多くの人に抵抗のあったキーボードから面倒な命令を打ち込むという操作から解放される。また、使い方を習得するために命令を記憶する必要もない。そして、そのような簡単な操作により得られる情報として、文書の他にもイメージや動画などの直感的に非常にわかりやすいものがまったく同じ操作で得られる、

また、もうひとつの普及の理由としてその情報提供の容易さがあげられる。情報を提供するためには、インターネットに接続され、HTTP Daemon プログラムが稼働しているサーバ計算機の中に、ホームページと呼ばれるテキストファイルを HTML 形式で作成すればよい。HTML 形式はもともとやさしいものであるし、最近ではワードプロセッサプログラムや表計算ソフトウェアからも直接この形式でファイル出力ができる。したがって、ほとんど自分の日常の仕事の中のファイルセーブくらいの労力で、情報提供が可能である。また、インターネットに接続することは、これまでの計算機ネットワークに較べて非常に容易であり、直接対等の立場で接続する計算機の了解さえとり、インターネットアドレスをもらえば、それ以上の手続きは必要ないし、費用もほとんどかからない。さらに、HTTP Daemon プログラムの設定も非常に容易である。また、インター

ネットプロバイダと呼ばれる接続業者の存在は個人のインターネットへのアクセスをきわめて容易にした。これらの理由により、インターネット上のホームページの量はきわめて短期間のうちに爆発的に増大した。その質に問題があるものも多いとはいえ、有用なものも多く、利用者も増え、それがさらにホームページの増加を招くという状態になっている。

利用者や情報の量が多くなるにつれて、その要求も多様になり、また高度になってきた。簡単な対話の利用を可能にするために、HTTP Daemon には当初から、CGI (Common Gateway Interface) 機能が組み込まれていた。これは HTML ファイルの送信を要求するかわりに、サーバ上のプログラムの起動とその実行結果の送信を要求するものである。ブラウザから CGI プログラムに引数を渡すこともできる。この機能を利用すれば、WWW のデータベース的な利用が簡単に実現できる。

WWW を頻繁に利用する人が増えて、その慣れたインタフェースで他の仕事も実行したいという要求がでるのは当然である。われわれも WWW ブラウザ上で操作できるような統計パッケージの GUI を作成したい。ただ CGI では入力 of 簡単な誤りでさえサーバにその入力を渡して、プログラムの実行が始まらなるとその誤りを発見できない。これではただでさえ容量不足であるネットワークが無駄な通信であふれてしまうことになる。そこでブラウザにおいても判断機能をもたせ、入力のチェックくらいはサーバに送るまえに実行するのが便利である。さらに複雑なこと、例えば統計パッケージの GUI をブラウザで実行するというようなこと、も行いたい。

このような目的のために、ブラウザに組み込まれた代表的なもの Java 言語による Applet 機能と JavaScript である。なお、本稿では主として UNIX OS 上でシステムを作成することを考えているので、そこで現在利用できない技術は考察しない。特に Microsoft Windows 上で用いら

れている WWW ブラウザ Microsoft Internet Explorer にはいくつかの注目すべき技術が装備されているが、本稿では考慮しない。

Java 言語による Applet 機能は Java 言語で記述されたプログラムを Java コンパイラで Java 中間言語に変換したものをネットワーク上でやりとりし、ブラウザの中に装備された Java 中間言語インタプリタが解釈、実行するという技術である。Java 言語は Applet 以外の一般的な用途に使うことのできるオブジェクト指向言語としても優れた設計がなされており、また、GUI の構成のための部品も完備しているので最近非常に注目を集めている。

JavaScript は代表的な WWW ブラウザの一つである Netscape Navigator の拡張言語であり、HTML のひとつの拡張として実現されている。文書内に埋め込まれた JavaScript プログラムを表示するかわりに Netscape Navigator の HTML インタプリタが解釈、実行するわけである。

これらの機能を実現する際にもっとも重要視しなければならないことのひとつはそのセキュリティである。WWW はネットサーフィンなどという言葉があるように、世界中の計算機の中を気の向くままに徘徊できることが特徴である。その中には悪意のあるものがあっても不思議ではない。したがって、受け取ったファイルに含まれているプログラムが自分の計算機の中で自由に動けるようではあまりに危険である。そのため、Java applet では現在のところ、送られてきたプログラムはローカルディスクのファイルにいったいアクセスできない。また、JavaScript にはそもそもファイルアクセスのための機能がない。

これらの制限はウイルスやワームと呼ばれる有害なプログラムが蔓延している現状をみれば理解できるものであるし、インターネット自身が電子ショッピングや電子マネーなどのように社会の基本インフラストラクチャとなりつつあることを考えれば、必要なものといえる。

しかしながら、われわれが考えているように WWW ブラウザ上で GUI を作成するためにはこれは致命的な欠点となる。特にローカルディスクからプログラムやデータを読み込む機能は不可欠である。

そこで、セキュリティのレベルを自分で変更できて、しかもブラウザの機能拡張として十分な能力を有するようなものが必要となる。そのために、われわれは Tclet を利用することにする。

4.3 Tclet

Tclet は Netscape Navigator の Tcl プラグインとして開発された Tcl/Tk のインタプリタを利用する Tcl/Tk プログラムである。プラグインは Netscape Navigator の機能拡張法の一つで、定められた形式で作成された実行ファイルを Netscape Navigator に統合する技術である。音声や動画などマルチメディア関係を中心に多くのプラグインが作成されており、無料で配布されていることが多い。Tcl プラグインは Sun Microsystems 社が開発しており、現在無料でそのベータ版が配布されている。Sun 社は最近、Ousterhout を中心とする SunScript group を設立し、Tcl/Tk をブラウザ上の標準スクリプト言語にすることを目標にしているようである。

Tclet プラグイン自身は完全な Tcl/Tk インタプリタの機能をもっているが、セキュリティ保護のために通常それに制限をかけている。それを Safe-Tcl と呼んでいる。ただ、Security Policy という機能があり、それを定義することにより、ユーザが制限をゆるめることができる。制限をゆるめればそれだけ外部から侵入する危険が増えるので、そのことはきちんと認識しておかなければならない。Sun 社でも Security Policy の自作には充分注意をはらうように、という警告を行っている。Sun 社自身からは現在、4 種の Security Policy が提供されている。それらは、ブラウザ

の他の部分とやりとりする機能をもたせた Browser, ローカルコンピュータ上にファイルを保存できる Tempfile, リモートサーバとソケットを通して通信できる Safesock, Tcl/Tk の全機能を利用できる Trusted, である。また, ドメイン名によって Security Policy ごとの許可不許可を指定できる。

われわれの GUI においては Browser Security Policy を用いるのが適当である。これにより, ブラウザの各部分とのやりとり, JavaScript の利用, 特定のディレクトリでのファイルの読み書きが可能となる。ただし, Tcl/Tk の命令の中には利用できないものもある。特に menu コマンドや toplevel が利用できないので, プルダウンメニューや新しいウィンドウの作成は不可能になり, 異なるやりかたを考えねばならない。具体的な実現法は 6 節で述べる。

5 Tcl/Tk による SHAZAM GUI

本節では混合ユーザインタフェースの考え方により Tcl/Tk で作成した SHAZAM の GUI について述べる。ここで作成したものでは SHAZAM プログラムと GUI プログラムは同一の計算機上で稼働する。

5.1 基本設計

SHAZAM プログラムは SHAZAM 言語のインタプリタであるから基本的な対話型操作は可能であるが, 先に述べたようにあまり使いにくい。そこで, Tcl/Tk により行入力のユーザインタフェースを作成し, プログラムの加工を行えるようにするとともに, SHAZAM からの出力についても編集ができるようにする。また, グラフの表示も直接できるようにする。さらに, プログラムの作成補助のための HELP 機能を充実させる。

基本設計としては以下のように考える。

まず、画面を大きく上下に2つに分け、上側を Program Window、下側を SHAZAM Window とする。下側の SHAZAM Window は普通の SHAZAM インタプリタ画面と同じである。ただし、画面のスクロール、画面のエディット、セーブおよびプリント、入力コマンドの履歴保存および繰り返し、が可能であるようにして、この Window だけでも対話が容易になるようにする。グラフィック機能に関しては SHAZAM の GNU-PLOT プログラム作成機能を利用するが、簡単な GNUPLOT エミュレータ機能を付加することにより、高精度グラフィックスをオプション GNU を加えるだけで自動的に表示できるようにする。なお、よく利用する命令の DEMO, STOP はボタンのクリックだけで実行できるようにする。

上側の Program Window は SHAZAM プログラムを作成するためのもので、その結果を SHAZAM インタプリタに送ることができる。まず、エディタの基本的な機能はすべて備える。ファイルのロード、セーブ、プリント、テキストのカット、コピー、貼りつけ、検索および置換が可能である。SHAZAM インタプリタとの通信機能として、ファイル全体、その一部または一行ずつを送ることができるようにする。

また、SHAZAM コマンド作成補助機能を備える。これは、コマンドの機能別分類メニューやアルファベット順メニューからの選択、コマンドの説明と引数のメニューからの入力および変数の選択、オプションの説明表示とメニューからの入力および選択を可能にする。この機能は SHAZAM の HELP コマンドを拡張するものである。

5.2 使用法

Tcl/Tk のスクリプトである shawin を実行することにより作業を開始する。UNIX の端末画面からはプロンプト (%) のあとに

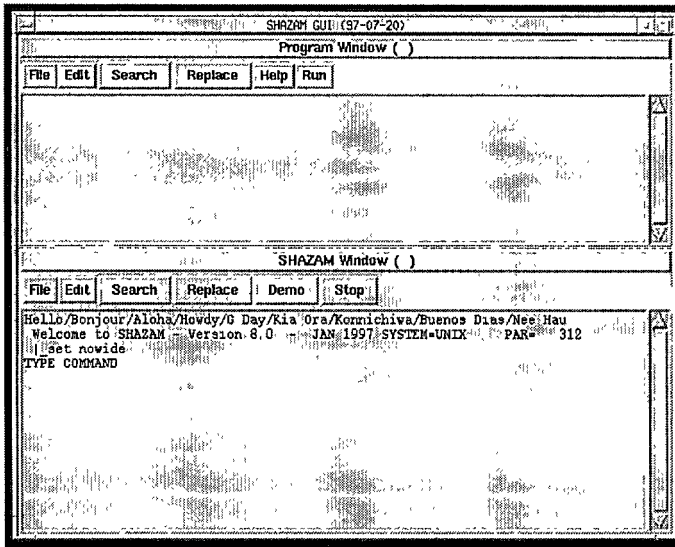


図 1: SHAZAM GUI の開始

% shawin

とすれば画面上にウィンドウが開き、同時に SHAZAM インタプリタプログラムが下側の SHAZAM Window で起動される（図 1）。

下側のウィンドウでは SHAZAM プログラムを直接利用しているときと同じ操作が可能である。すなわち、SHAZAM プログラムからのプロンプト TYPE COMMAND のあとに任意の SHAZAM コマンドを入力し、リターンキーを押せば、それが実行される。

このウィンドウ内のカーソル（縦棒型）はカーソルキーおよびマウスにより移動することができる。そして、普通のエディタとほぼ同じことが可能である。たとえば、シフトキーとカーソルキーの同時操作により、またはマウスのドラッグにより範囲の選択ができる。そして Edit ボタンによ

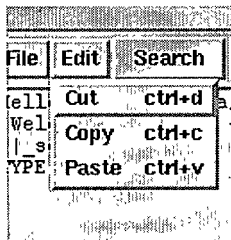


図 2: Edit ボタンのメニュー

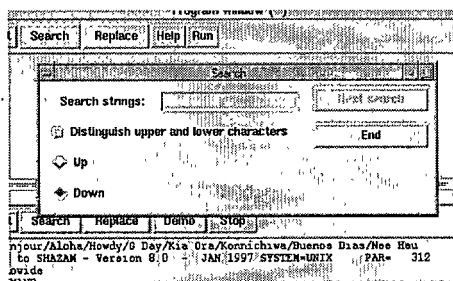


図 3: Search ボタンによる入力ウィンドウ

り、またはそこに表示されるショートカットキーにより、Cut, Copy, Paste ができる (図 2)。

Cut は選択された範囲のバッファへの移動, Copy は選択された範囲のバッファへのコピー, Paste はバッファの内容のカーソル位置への挿入である。また, Search, Replace ボタンをクリックすることにより、それぞれの指示を与えるためのウィンドウが開くから、それにより検索、置換を行うことができる (図 3)。

リターンキーの働きは通常のエディタと異なる。まず, SHAZAM プロンプト TYPE COMMAND の次に SHAZAM コマンドを入力したあとでリターンキーを押せば、そのコマンドを SHAZAM プログラムに渡し実

行する。それ以外の位置にカーソルがあるときにリターンキーを押すことは、そのカーソルを SHAZAM プロンプトの直後に移動するだけである。改行コードの挿入はできない。また、このウィンドウ上から SHAZAM に送られたすべてのコマンドは保存されている。それを呼び出すためにはカーソルが SHAZAM プロンプトの直後にあるときに、ctrl-P（コントロールキーを押しながら P キーを押すこと）を押せばよい。コマンドを過去のものに順次さかのぼって呼び出せる。また、途中で ctrl-N によって新しいコマンドを順次呼び出すこともできる。これらの機能により呼び出したコマンドはカーソルキーなどを用いて一部修正したあと、リターンキーで SHAZAM プログラムに送ることになる。

Demo ボタンは DEMO コマンドを、Stop ボタンは STOP コマンドを SHAZAM プログラムに送るものである。STOP コマンドを送ると SHAZAM プログラムが終了すると同時にこのインタフェースウィンドウも消滅する。

File ボタンをクリックするとプルダウンメニューがあらわれる（図 4）。

Clear はウィンドウの内容を消去する。Save はこのウィンドウの内容を以前のファイルに上書きする。ファイル名が指定されていなければ、そ

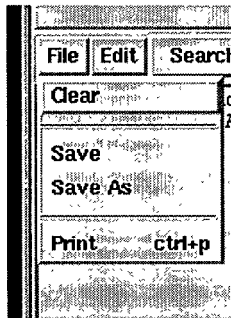


図 4: 下側の File ボタンのメニュー

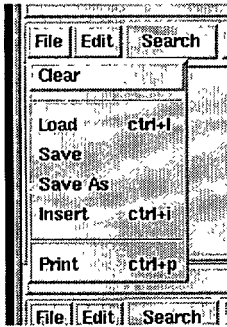


図 5: 上側の File ボタンのメニュー

れを指定するようにうながすウィンドウがあらわれる。なお、現在考察しているファイル名は下側のウィンドウのタイトルバーの括弧のなかに表示されている。Save as は新しいファイル名でこのウィンドウの内容をセーブする。Print はこの内容をプリンタに送る。そのときプリンタ名を指定するようにうながされる。

このインタフェースプログラムにおいて、下側のウィンドウでこのように対話的に SHAZAM を利用するのも便利であるが、長いプログラムを作成するには上側の Program Window を用いるのがよい。このウィンドウは基本的にエディタである。Edit, Search, Replace ボタンは下側のウィンドウのものとまったく同じである。File ボタンのメニューにはファイルを読み込むための Load, Insert が追加されている（図 5）。また、リターンキーはどの場所にあっても改行コードの挿入を意味する。

このウィンドウで作成したプログラムを下側のウィンドウで稼働している SHAZAM プログラムに送るためには、Run ボタンを用いる（図 6）。このボタンを押した場合、3つのメニューがでてくる。All はこのウィンドウの内容のすべてを送る。Selected はマウスあるいはシフトとカーソルキーで選択された範囲を送る。この場合、選択された範囲は連続した行

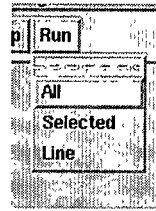


図 6: Run ボタンのメニュー

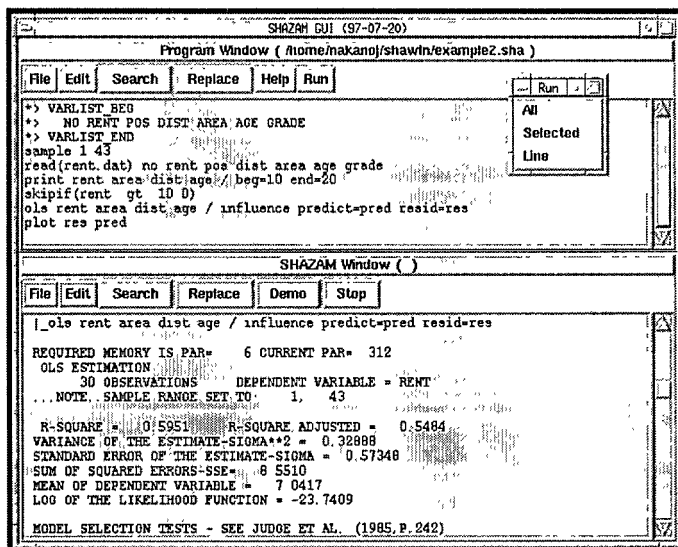


図 7: 切り離された Run ボタンのメニューおよび実行例から構成されていなければならない。最初が行の途中から始まったり、最後の行の途中で終わったりすることは避けねばならない。line はカーソルのある行を SHAZAM に送る。そのあと、カーソルは次の行の先頭に移動する。したがって line を押し続けることによって行を次々に実行していくことができる。なお、Run ボタンメニューの上部の破線部分をクリックすればメニューを切り離して画面上に置いておくことができる。このメニュー切り離し機能は破線部分のあるすべてのメニューで有効である。図 7 は、3 節の example1.sha と同じ計算を行う example2.sha というプログラムを読み込んで、実行した結果である。

Help ボタンを押すと 3 つのメニューがでてくる (図 8)。最初の Editor Usage でこのエディタの操作法の簡単な説明が見られる。次の二つは SHAZAM コマンドの作成補助機能を起動するためのものである。Com-

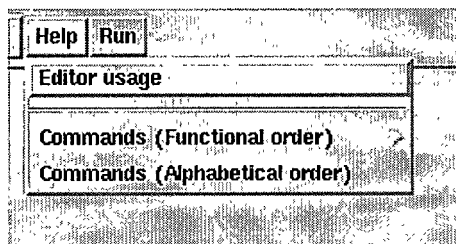


図 8: Help ボタンのメニュー

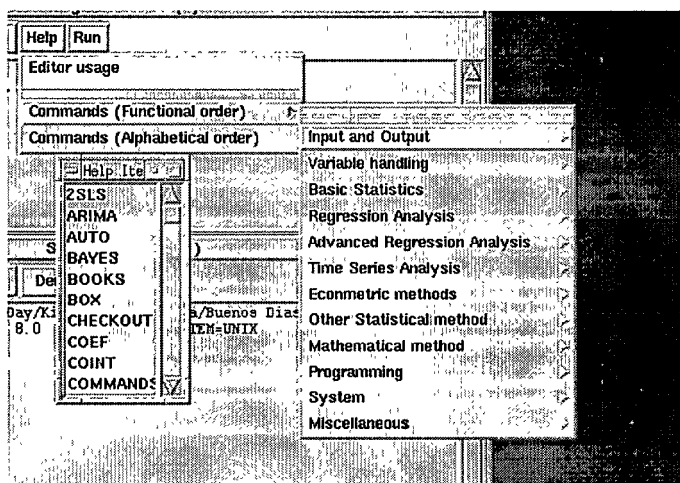


図 9: Help(Commands) のメニュー

mands (Functional order) では機能別に分類されたコマンド名で, Commands (Alphabetical order) ではアルファベット順に並べられたコマンド名で, 説明がほしいコマンドを指定できる. 指定されたコマンドのヘルプウィンドウが別ウィンドウとして開き (図 9), そこでコマンドをマウスのクリックで構成することができる.

ヘルプウィンドウの例として OLS を見る (図 10). このウィンドウは

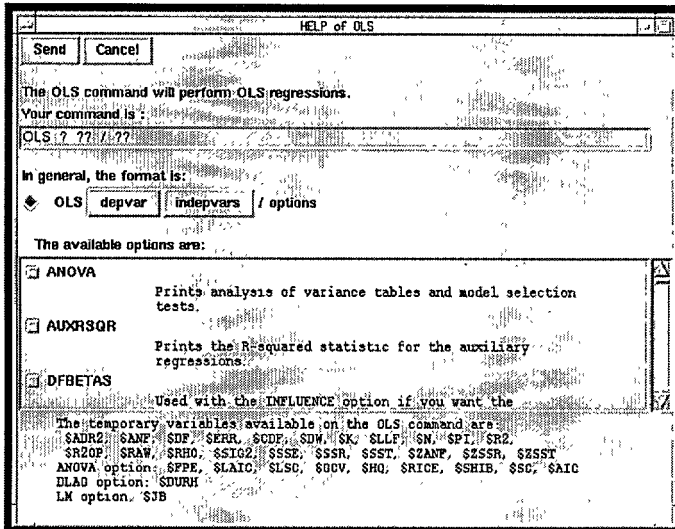


図 10: Help(OLS) のウィンドウ

上のほうから、ボタンエリア (Send および Cancel ボタン)、コマンドの概略説明、ユーザの作成中のコマンド、コマンドの形式、オプション指定、その他、の構成になっている。コマンドによってはこのうちのいくつかの部分が省略されていることもある。コマンドを作成するには、変数指定の部分はコマンドの形式部の対応するボタンをクリックすれば、変数名の選択のためのウィンドウが開く (図 11)。それまでに指定した変数名は記録されているので、クリックして選択し、Select ボタンを押せばよい。

この図においては example2.sha で利用されている変数名がリストされている。これは example2.sha の最初の部分に定められたコメントの形式

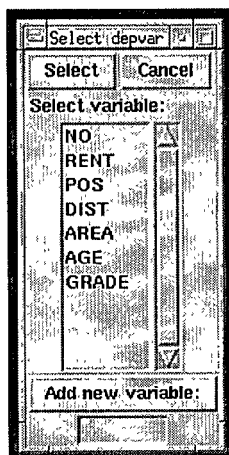


図 11: 変数名の選択のためのウィンドウ

で記述されているものから読み取られる (図 7)。なお、このような使用されている変数名に関するコメントはファイルセーブの際に自動的に付加される。

新しい変数は一度 Add new variable: のところに入力し、そのボタンを押すことで変数リストに登録し、Select ボタンを押す。このように使用する変数をすべて登録しておくことにより、以後はマウスで選択するだけで入力されるので、入力ミスがなくなる。選択された変数は Your command is: の正しい位置に挿入される。この部分には最初のうち、? や ?? という記号が書かれているが、これは正しいものを選択しなければならない。? の部分には一つの、?? の部分には一つ以上のものが書ける。また、変数選択の場所でボタンの表示の最後が var となっているものは一つの

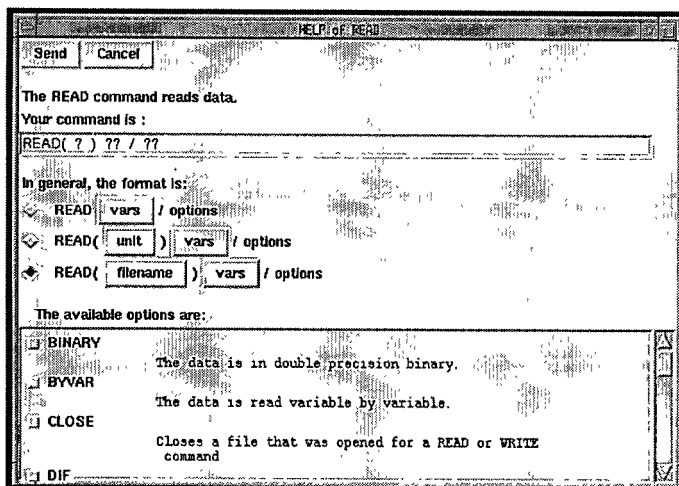


図 12: Help(READ) のウィンドウ

変数名を, vars となっているものは一つ以上の変数名を選択できる (OLS では depvar と indepvars).

オプションに関しては, オプション名だけのものはそれをクリックすれば選択され, もう一度クリックすれば消される. オプション名のあとに等号があるものは等号のあとに変数名や数字などを入れなければならないので, そのためのウィンドウが開かれる.

また, コマンドの中には複数の異なる形式で利用されるものもあり (例えば READ コマンド. 図 12), そのような場合にはコマンドの形式のところで利用したい形式を選択すればよい. このようにしてコマンドの全体を正しいものにしたあとで, 最上部の Send ボタンを押せばそれが Program Window に送られる. このヘルプウィンドウを用いれば, 新しい変数名や数値などを入力することを除いて, ほとんどの操作がマウスのクリックで可能となり, GUI といえるものになる.

5.3 実装

GUI プログラムのプロセスは、パイプ機能を介して GUI プログラムから自動的に起動される SHAZAM プログラムのプロセスと通信を行う。GUI プログラムから SHAZAM の命令を SHAZAM プログラムに送ることは Tcl 言語の機能を利用すれば簡単であるが、SHAZAM プログラムからの出力を表示するためには工夫を要する。SHAZAM プログラムの出力は、すべて一時ファイルに送られる。SHAZAM に命令を送ったあと、GUI プログラムは定期的（デフォルトでは 1 秒ごと）にこのファイルを見に行き、新しい行が増えていれば、それを読みとり、必要な部分を下側の SHAZAM Window に表示する。命令の終了を検知するために、GUI プログラムは利用者が指定した命令の最後に特定の文字列を表示する命令を付加して SHAZAM に送る。GUI プログラムはその文字列を読みとった場合に、命令に対する出力が終了したことがわかるので、以後一時ファイルを見に行くことを中止する。もちろん、この特定の文字列を SHAZAM Window に表示することはしない。この実装では、対話型の利用の際、応答が若干遅れるという欠点があるが、実用上の障害とはならないだろう。

エディタ機能の実現においては Tcl/Tk のウィジェットに組み込まれている機能とバインド機能を利用すれば比較的簡単に実現できる。ここでのプログラムでは宮田・芳賀（1995）のエディタプログラムを利用した。

コマンド作成補助機能の実現では、SHAZAM に備わっている HELP 機能をできるだけ利用することにする。SHAZAM の HELP 命令は“HELP コマンド名”とすることにより、そのコマンドの簡単な説明を表示するものである。その説明の中では、命令の概略、形式、利用できるオプションとその説明の一覧、一時変数のリスト、などがほぼ同一の形式で表示される。この内容は SHAZAM インタプリタプログラムとは別の

SHAHELP というテキスト形式のファイルにすべて格納されている。そこでわれわれもこのファイルを利用する。ただし、このファイルはテキストとしての表示だけを考慮して作成されているので、フォーマットが必ずしも厳密に統一されているわけではない。そこで、われわれはこのファイルをできるだけ利用するが、形式がほかのものと異なる場合には若干の書き直しを行うことにする。SHAHELP の形式として、あるコマンドの説明は“+コマンド名”で始まる行から始まり、“-コマンド名”という行で終る。また、“-コマンド名”の行の直前の行は“?STOP”である。このため、あるコマンドに関する部分を取りだすことは非常に簡単である。

説明の順番は、まず、等号ではじまる概略説明がある。次に、“In general, the format is:”で始まることの多いコマンド形式部が続く。それに続いてオプションの説明が始まるがそれは“The available options are:”というような文章で始まることが多い。オプションの説明はオプション名の後にインデントしてその説明を書く、という形式がだいたい守られている。そしてそのあといくらかの補足説明が加えられる。したがって、SHAHELP から一行づつ読み込み、これらの規則に照らし合わせながらヘルプウィンドウ上に必要な Tcl/Tk の GUI 部品を埋め込んで行けば良い。

なお、Help ボタンの Comannnds (Functional order) メニューの内容は SHAHELP.LST というファイルに整理したものを読み込むことにする。

6 WWW ブラウザ上の SHAZAM GUI

この節では、WWW ブラウザの GUI を利用して、SHAZAM を利用することを考える。前節とは違って、GUI と SHAZAM インタプリタは別の計算機上のプロセスであることが普通である。

6.1 基本設計

WWW ブラウザを利用することにより、GUI にはいくつかの特徴と制限が与えられる。このような GUI を実現するためにはブラウザの機能拡張が可能でなければならない。それが充分なものであれば GUI は OS よりもブラウザに強く依存する。したがって、ブラウザが複数の OS 上で同等に動いていればそれらすべての OS 上で利用できることになる。また、GUI の基本的な使用法はブラウザのものになる。したがって新たに使用法を覚えるという抵抗感が少ない。さらに、SHAZAM インタプリタはサーバ計算機にあるので、ユーザが SHAZAM インタプリタのバージョンアップなどのメンテナンスを行う必要がない。これは以前のメインフレーム計算機と同様の集中管理ができることを意味する。ただ、その集中管理すべき計算機が自由に分散できるところが大きな違いである。最後に WWW の性質上、作業は基本的にバッチ処理になる。サーバとブラウザがつねに接続しており、作業を対話的に行うという形式には向いていない。

ここでの GUI は以上のような特徴を考慮しながら、前節で作成した GUI をブラウザ上で稼働するようにしたものである。ただし、先に述べたようにネットワークに関するプログラムでは、セキュリティ保護を考慮しなければならない。したがって、ローカルディスク上のファイルのアクセスには制限をもうけざるを得ない。ここでは特定のディレクトリへの読み書きだけを許し、また、ファイルのリストを取るようなことは許さないことにする。

WWW を利用するので、サーバファイル上の SHAZAM プログラムの起動には CGI 機能を利用する。したがって CGI プログラムで渡された SHAZAM プログラムを加工することができる。これを利用して SHAZAM にネットワーク対応させたコマンドを付加することにする（中野, White 1997). それにより SHAZAM プログラムの分散処理も可能に

なる。

6.2 操作法

本節での GUI は Netscape Navigator (バージョン 3.01 以上) で利用できる。いくつかの UNIX 系 OS (Linux, Solaris) と Microsoft Windows 95 で動作することを確認している。ただし、Tclet プラグインがまだ正式バージョンでないこともあってか、使用中に不安定になることもある。

本 GUI を使用するには最初に一回だけ、プラグインのインストールをしなければならない。プラグインのアーカイブファイルをダウンロードし、それを解凍する。そしてそれぞれの計算機に固有の方法でインストールする。その方法に関しては Sun Microsystems 社の WWW ページを参照すること。そして、SHAZAM サーバのドメイン名を Security Policy のコントロールファイルに加える。例えば Linux の Netscape では Netscape のインストールディレクトリの中の tcplug/2.0/tcl8.0/policies/browser.data を変更すればすべてのユーザが利用できる。

また、SHAZAM サーバとなる計算機では SHAZAM プログラムとともに CGI プログラムや HTML ファイルをアクセス可能にしておかなくてはならない。われわれが作成した CGI プログラムは単に SHAZAM プログラムを実行できるだけでなく、実行 SHAZAM サーバの指定、ファイル読み込み時に URL (Universal Resource Locator) を指定できる、など WWW を利用した分散処理機能を付け加えてある。これらのインストールに関しては配布ファイルを参照してほしい。

以上の準備が終わったあとで、SHAZAM インタフェースページをブラウザで指定すると、前節とよく似た、上下に 2 分割されたページがあらわれる (図 13)。この図は前節の example2.sha とほとんど同じプログラム ex-

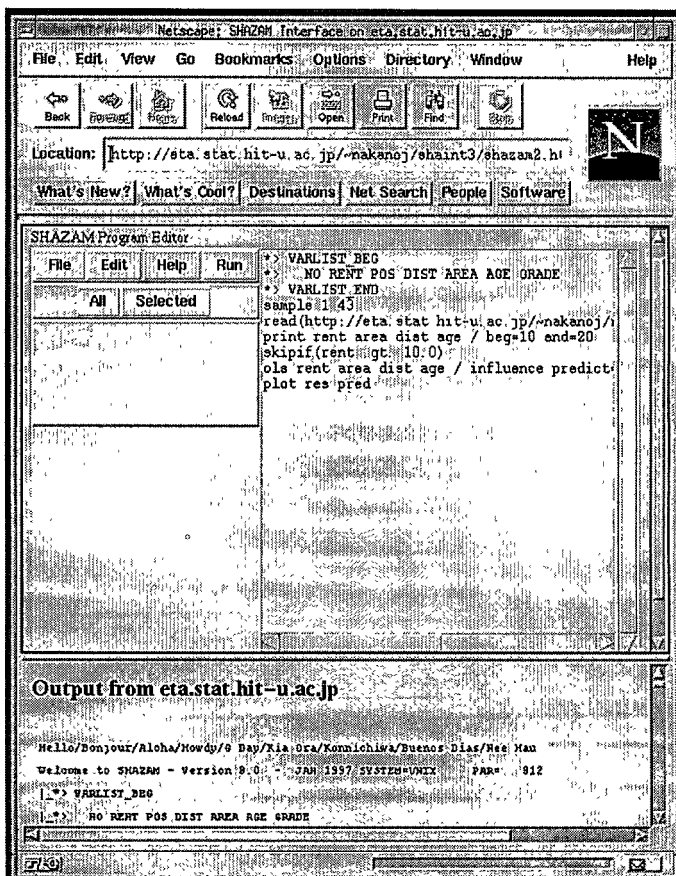


図 13: WWW インタフェースの実行例

ample3.sha を実行した直後の例である。

下側は SHAZAM 出力を表示するページである。その出力が計算されたホスト名を太字で示したあと出力を表示する。これはこの GUI で書ける SHAZAM プログラムが WWW 用に拡張されたものであるため、プログラムを実行するホスト名を指定することができるのでその場合の出力元をはっきりさせるためである。このウィンドウの内容をローカルファイルに出力するにはこのフレームを指定してブラウザの File ボタン、Save Frame as…の機能を利用する。

上側のウィンドウは SHAZAM プログラム編集のためのウィンドウである。大きく左右に別れており、右側がエディットのためのウィンドウ、左側がボタンや表示、入力のためのウィンドウである。

右側のウィンドウはエディタの基本的機能を備えている。矢印キーによるカーソルの移動、マウスやシフトキーを用いた範囲の選択、その部分のカット、コピー、および張り付けができる。また、検索および置換も可能である。これらは左側のウィンドウの最上部のボタンのひとつである Edit ボタンを押し、2 番目の部分に Cut, Copy, Paste, Search, Replace のボタンを出し、それらをクリックすることにより実行される。Search と Replace に関しては上から 3 番目の部分に入力のためのウィンドウが開くので、そこで必要な情報を入力する (図 14)。このように、必要なボタンや入力ウィンドウはすべて左側の部分に階層的に下に向かって表示される。このとき、左右のウィンドウの大きさは自動的に調整される。左側のウィンドウの最上部のボタンは常に表示されている。

File ボタンを押すと Clear, Load, Save ボタンが 2 番目のメニューとしてあらわれる。Clear は右側のウィンドウの内容を消去する。Load および Save はそれぞれローカルファイルの読み書きを行う (図 15)。

ただし、読み書きできるディレクトリに関しては制限がある。それは

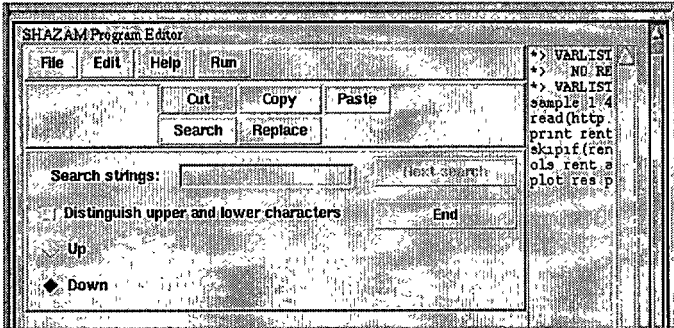


図 14: Search のウィンドウ

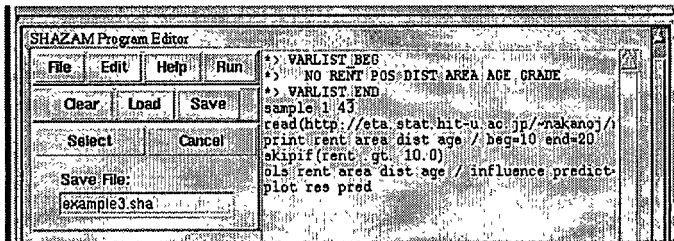


図 15: Save のウィンドウ

Browser Security Policy のコントロールファイルで指定されたディレクトリ内にブラウザによって作成・指定されたディレクトリだけにアクセスできるというものである。そのディレクトリの位置はもとの SHAZAM インタフェースページの URL と関連づけられており、そのページにいつアクセスしても同じディレクトリが使用される。そしてユーザがそのディレクトリを前もって最後まで指定することはできない。例えば、Linux においてはホームディレクトリの下 `browserTmp/private` というディレクトリの中に `dir0`, `dir1`, `dir2`, `dir3`, ... のようなディレクトリが作成され、このうちのひとつが使用される。どれが使われることになるかは、はじめ

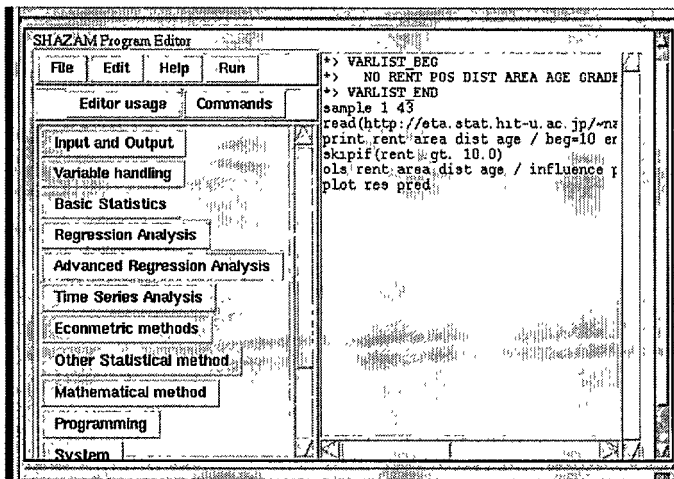


図 16: Help (Commands) のウィンドウ

てローカルファイルにアクセスして使用するまでわからない。したがって、ユーザは自分でどこが使用されているかを調べねばならない。また Save や Load の入力にディレクトリ名は付加できない。この仕様はやはりセキュリティ保護のために作成された制限である。

最上部の Help ボタンを押すと Editor usage と Commands の 2 つのメニューがあらわれる。Commands は前節の Functional order のものと同じであり、押すと機能別に分類されたリストがでる (図 16) ので、機能、コマンド名の順に選択する。すると前節と同じ形式のヘルプウィンドウが左側上から 3 番目に開く (図 17) ので、必要なものを選択すればよい。

最上部の Run ボタンを押すと All, Selected の 2 つのボタンがあらわれる。前節と異なり、対話的な処理ができないので、一行ずつプログラムを

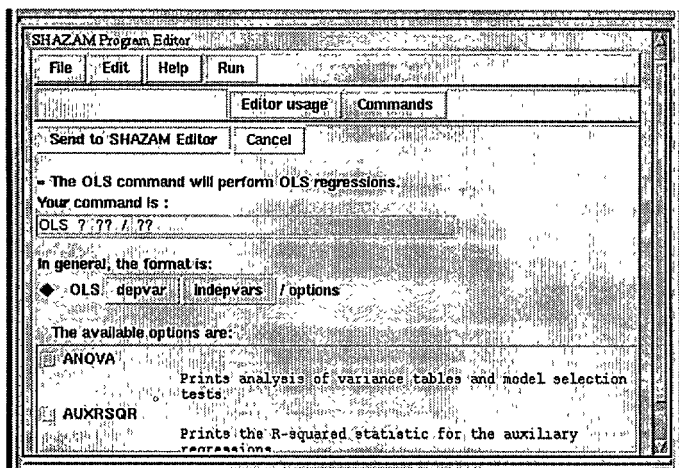


図 17: Help (OLS) のウィンドウ

渡すことはできない。また、そのためにたいていの場合、All が利用されるであろう。

6.3 実装

SHAZAM サーバ側の CGI 機能の実現については中野, White (1997) にくわしい。プログラム example3.sha では READ コマンドで URL を記述しているが、これを処理するのはこの CGI プログラムである。ただこの論文を作成した時期の Tcl/Tk は TCP/IP 関係の機能が完備していなかったのでそれが完備した拡張言語である TclX を利用したが、現在では Tcl/Tk 自身にこれらの機能があるので TclX は利用していない。また、グラフ出力については GNUPLOT やグラフ形式変換プログラムを利用して高精度グラフィックスをブラウザ上に表示できるようにしてある。

SHAZAM インタフェースのページは HTML の FRAME 機能を用いて上下に分割し、上側のフレームに Tclet をロードする。Tclet のプログラムは前節のものを基本にしてメニューの部分を書き直す。Safe-Tcl ではメニューバーと新しいウィンドウの作成ができないので、すべて左側のウィンドウに埋め込むようにする。その際、以前に利用したものを消去し、新しいものを上書きするという方法をとる。

エディタのプログラムを CGI に送るには Tclet から JavaScript を呼ぶようにして JavaScript の submit 命令を用いる。これは Browser Security Policy で可能となる機能である。

現在のところヘルプの内容を記述した SHAHELP ファイルとコマンド名を機能別に分類した SHAHELP. LST ファイルを Tclet が読み書きできるディレクトリに最初にコピーしておかなければならない。これは少々面倒なので、改善を要する点である。

7 おわりに

パーソナルコンピュータの性能の高度化と価格の低下によってその普及はめざましい。現在ではそれらには最初から GUI を基本とした OS が附属しているので、CUI をまったく知らない利用者が普通になっている。また、統計パッケージの普及も進んでおり、ますます多くの初心者が利用するようになっていく。明らかに統計パッケージのユーザインタフェースの GUI 化は避けられない状況である。

先に述べたように統計パッケージからプログラミングの必要性がなくなるとは思えない。しかしプログラミングの多くの機能を、できればすべての機能を、GUI から利用できるようにしたいという利用者が多くなっているのである。

本稿で作成した GUI は CUI を使い易くするという方向での混合ユーザインタフェースであった。これでは中心はあくまで CUI ということになってしまう。SHAZAM が設計された時点では CUI しか利用できなかったのであるからこれはしかたがないといえるし、ここで作成した GUI が CUI だけの場合より初心者を使い易いことは確かである。ただ、このような CUI 中心の混合ユーザインタフェースは少々不自然なものに見えるかもしれない。必要とされているものは GUI を中心として CUI も備えるという形の混合インタフェースであろう。このためには統計パッケージの設計を GUI 中心にして行わなければならない。そのような統計パッケージは現在はまだあまりないと思うが、今後研究が必要であろう。本稿で利用したようなツールを用いることにより、計算機の専門家ではない統計学者自身がそのような新しい統計パッケージを作成することが可能になってきたと思われる。

また、インターネットの利用もますます盛んになるであろう。現在、WWW に関する技術は急速に発展している。WWW の基本が非常に柔軟に設計されていたので、多くの機能を組み込むことができたからである。しかし、利用者の要求はとどまることを知らないの、現在の枠組で対応しきれなくなる時期もそう遠いものではないだろう。WWW の基本設計自身が拡張されるようになっていられる。そのような場合に対して、統計パッケージの側からも必要な機能を研究しておく必要がある。本稿のように現在の WWW ブラウザを利用する研究は GUI の標準化という点で重要と思うが、それ以外に、独自のインターネット利用法を研究することも必要であろう。本稿でふれた最新技術である Java 言語や Tcl/Tk 言語は WWW 以外のネットワークの利用も非常に容易に行えるようになっている。したがって、これらを利用して新しい形のネットワーク対応統計パッケージを作成することが可能であろう。

参考文献

- [1] Chambers, J. M. and Hastie, T. J. (ed.) (1992). *Statistical Models in S*, Wadsworth & Brooks/Cole, Pacific Grove, Calif. (邦訳：柴田里程訳. *S と統計モデル—データ科学の新しい波*. 共立出版, 東京)
- [2] Liu, L. -M., Chan, K. -K., Montgomery, A. L. and Muller, M. E. (1995). A system-independent graphical user interface for statistical software, *Computational Statistics & Data Analysis*, 19 (1), 23-44.
- [3] 宮田重明, 芳賀敏彦 (1995). *Tcl/Tk プログラミング入門*. オーム社, 東京.
- [4] 中野純司, 山本由和, 岡田雅史 (1991). 知識ベース重回帰分析支援システム, *応用統計学* 20, pp. 11-23.
- [5] 中野純司, White, K. J. (1997). 統計パッケージ SHAZAM の WWW インタフェース, *統計数理*, 45 (1), pp. 41-47.
- [6] Ousterhout, J. K. (1994). *Tcl and the Tk Toolkit*, Addison-Wesley, Reading. (邦訳：西中芳幸, 石曾根信訳. *Tcl & Tk ツールキット*. ソフトバンク, 東京)
- [7] White, K. J. (1997). *SHAZAM User's Reference Manual Version 8.0*, McGraw-Hill, New York.
- [8] 山本由和, 中野純司, 田村義保 (1995). MS-Windows 版 TIMSAC72 の UNIX への移植, *統計数理*, 43 (2), pp. 285-292.