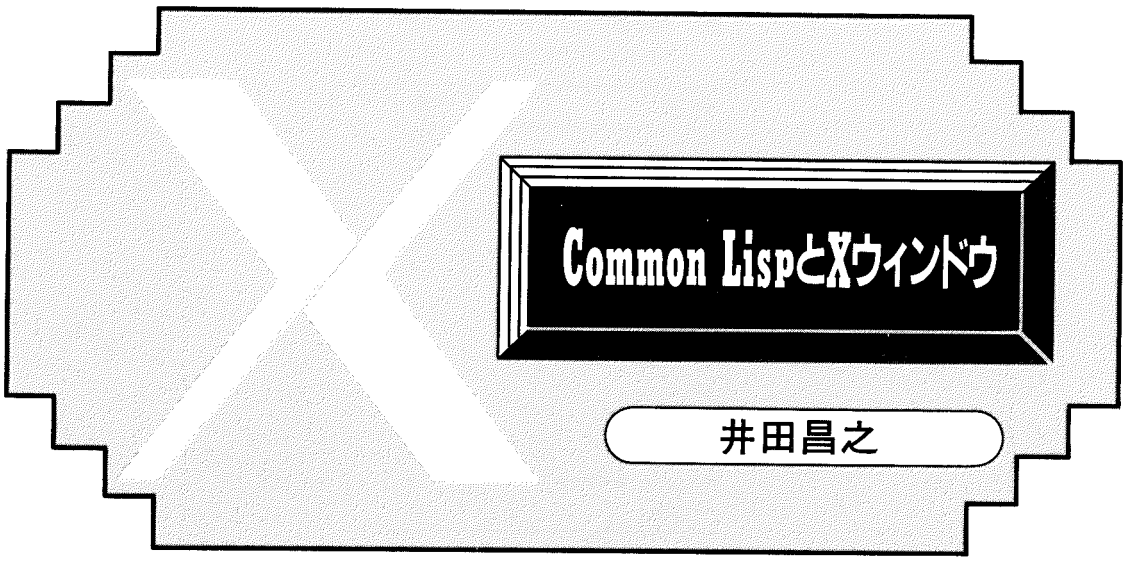


1. はじめに

1986年に Common Lisp が成立して以来，オブジェクト指向機能をはじめとしてさまざまな機能に対する集中的な検討が進められてきた．ウィンドウシステムあるいは GUI もその一つである．当初は Intellicorp 社の Common Windows の外部仕様を共通案とし，それを肉付けする形でさまざまな仕様がでるだろうと予想されていた．しかし，ウィンドウシステム/GUI 関連の全体の動き，技術の進歩と共に，ツールの仕様設定も UIMS レベル，ウィンドウツールキットレベル，インタフェースビルダ，ウィジェットライブラリ指向，などに分かれるようになってきた．現在では，商用・研究用に多数の仕様と処理系が存在する．それらの切口は同一ではなく，比較をするのも困難である．Lisp の特質がさらにそれに加わる．また，汎用の OS のもとで用いられる場合もあるが，同時に Lisp は，完全な OS レベルの機能をもった使い方もされる専用マシンが実用になった代表的な成功例でもあるので，そうした背景で機能設計がされるケースもある．汎用 OS のもとで利用される場合には Lisp 独自のものだけでなく，OS で標準的に用いられる GUI とのインタフェース，あるいはそれの使用感覚との共通化が図られたりしている．X ウィンドウ上のツールもこうした背景で分類できる．

2. どんなツールがあるか?

2.1 電子協でのアンケート集計の結果

X ウィンドウだけでなく，全般にどんなツールが使われているのか/使われるのかを知る調査の一例として，1990年度に電子協（電子工業振興協会）の処理系言語技術専門委員会（Lisp）で行なった Lisp での

表1　電子協でのアンケート集計結果
（ウィンドウツールキット）より

名　称	件数
CLX	29
MS-Windows	20
Presentation Manager	12
Common Windows	10
CLUE	4
YyonX	3
CLIM	2
LispView	1
Winterp	1
GINA	0
GARNET	0
Experss Window	0
その他	32

GUI/ウィンドウツールキットの利用に関する統計調査の結果を**表1**に示す．これによると，さまざまなツールが利用されていることがわかる．むしろ，まだまだこれからの分野といえなくもない．

なお，この表は実際に使用しているという回答を集計している．（この調査の回答者のほとんどは実際のLisp ユーザである．）回答総数は222件である（複数回答あり，またこの項目に対する無回答もある）．その他（32）の中には Symbolics Genera を始め，同一のツールに対する複数回答もあると思われるが詳細は不明である．

また，その処理系言語技術専門委員会のユーザインタフェースワーキンググループでは次のものをフォローし，共通例題（grapher）による実験的なコーディングを行なったりしている．（それらについては電子協の報告書にまとめられているが，ソースコードは `ftp.csrl.aoyama.ac.jp` の `graphers` の下にある．）

CLX, CLIM, CLUE, Common Windows, Express Window, LispView, Winterp, *YyonX*

これらは，Steve Strassman 氏（当時 MIT MediaLab., 現在は Apple Cambridge Lab.）が1990年春にまとめた9ツール（CLX, CLIM, CLUE, CLM-GINA, EW, Garnet, SOLO, Winterp, *YyonX*）のサーベイリストに基づいて，その中で日本国内で使えるものを取り上げたものである．読者のために，それらを簡単に紹介しておく．

1. CLX：Xlib 対応，X 11 のリリースに含まれている．
2. CLIM：Symbolics（Dynamic Windows）の機能に対応，ILA 社の開発．
3. CLUE：Xt level with CLOS, TI（テキサス・インスツルメンツ）による．
4. CLM-GINA：Motif toolkit，ドイツでの開発．
5. EW（Express Window）：Dynamic Windows on UNIX without CLOS．EW は Liszt Programming Inc. という会社により商品としての流通が最初試みられ，ついで PDS として扱われるようになっている．（`ftp.csrl.aoyama.ac.jp` に置かれている．）
6. Garnet：研究用，カーネギーメロン大学で開発．
7. SOLO（LispView）：OPENLOOK toolkit．最初は SOLO と呼ばれ，商品化に伴って LispView に

名前を変えた．Sun Microsystems による．

8. Winterp：Motif widget interpreter，X11 のリリースに含まれている．
9. *YYonX*：研究用，青山学院大学で開発．

3. どれを選ぼう？

X ウィンドウ上のさまざまな Common Lisp 処理系がもつ GUI 関連機能は処理系側の立場で見て，次のように分類できる．

1) 基本的には何もない．
1.1) GUI の具備は利用者に任せる．
1.2) Xlib 相当の機能を実現してくれる CLX を移植／用意しておく．
2) CLX だけでは「機能的に弱い」ので，もう少しハイレベルの道具として CLUE を移植しておく．（この延長として，さらに CLIO を移植する．）
3) そもそも KEE のウィンドウシステムであった Common Windows はその文脈での共通性も高いし，利用者も多いので，その機能を X 上にもってくる（Allegro Common Windows など）．
4) Lisp マシンのウィンドウシステムは革新的な機能をもっている．Lisp マシン上のソフトウェアを X 上に移植するには，その革新的なウィンドウシステムが不可欠である．と同時に，新しくソフトウェアを開発する場合にもその仕組みは大きな福音である．これを用意する（CLIM など）．
5) 現在，標準的だと考えられている look and feel がある．Motif と OPENLOOK である．Lisp の世界でもこれらとの共通性をもたせるのがよい（CLM, LispView など）．
6) 研究の目的のためのシステムなので，それが重要であり，特定の API との共通性に固執するわけではない．現在は独自の API の設定をしている（*YY*, Picasso, Garnet など）．
7) 独自の考え方に基づいて，さまざまなプラットフォーム上のアプリケーションポータビリティを追求している．したがって，GUI も独自のものである（LispWorks など）．

また，これらに加えて，Lisp 的な Interface Builder もある．それらでは，できあがったインタフェースは Lisp のソースコードとして出力される．

以下では，上記の1）から7）の概要を紹介すると同時に，概念的にいって最もきわだって対比できる二つの考え方を紹介する．Widget ベース思考と Presentation ベース思考である．

4. Widget ベースか Presentation ベースか？

Widget ベースか Presentation ベースか？ Lisp で書かれたプログラムの GUI を考えるときに大きな流れを作っているこの二つの間の違いを知ることは重要である．実際に使う上ではどちらの考え方に立っても扱えるし，また Widget ベースの上に Presentation を置いたり，Presentation ベースの上に Widget を見せることもできる．ここではその基本的な考え方について触れてみたい．Widget ベースの仕組みは他の章で扱われている仕組みと共通性が高いのでそちらに譲り，Presentation ベース思考の説明に重点を置く．

4.1 既存標準部品の組合せか，生きたオブジェクトの集まりか?

4.1.1 標準部品型ポップアップメニューの実現

ポップアップメニューを考えてみる．ポップアップメニューの動作はおよそ次のようなものである．

0. メニューの大きさ，起動方法，メニュー中の項目の並び，項目の選択によって呼び出す機能の指定などをあらかじめ設定しておく．
1. メニューをポップアップするシンボルなどをクリックし，（もしくは他の手段で）メニューを表示する．
2. 利用者はポップアップされたメニューの上でマウスを動かす．
3. 選択可能項目の上を通過しているときには，それが選択可能であることを示すためにハイライトされるなどの応答が起こる．「選択可能項目」とは，ふつうそれを示すパターンが記された「領域」のことである．どんなパターン，シンボルが表示されているかは人間の見やすさを助けるためであって，マウスセンス対象そのものには関係がない．この点に重要な性質がある．
4. 選択したい項目が見つかり，その上にマウスカーソルがきたとき，「これ！」という指示のつもりでボタンクリックなどをする．
5. クリックされた領域に対してあらかじめ関連づけられた機能が呼び出される．

図1 のようなものである．マウスカーソルがどの領域に入っているかということで，その領域にアサインされている項目を選択したと見なす．

Widget ベースで扱うという場合，こうしたことを可能とするためのポップアップメニューの部品が用意され，利用者はその機能を利用する．

4.1.2 Presentation によるメニューの実現

Presentation により，いろいろな Widget を実現することは容易である．もちろん専用の部品を前もって作っておくほうが性能のよい場合もある．また，Presentation でしかできにくいような柔軟な仕掛けもある．

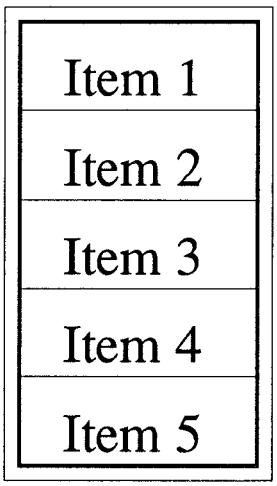


図1　領域指定によるメニュー

一例として，Presentation によるメニューを考えてみる．その場合の動作は次のようなものである．これを一般的なポップアップメニュー動作と比較してほしい．

0. 選択対象となるオブジェクトを表わすシンボル／アイコンをあるウィンドウに表示しておく．(各項目の大きさ，位置などには必ずしもこだわらなくてよい．)
1. そのウィンドウ上でマウスを動かす．
2. マウスセンス可能な「オブジェクト」上を通過するときには，それが選択可能であることを示すためにハイライトなどの応答が起こる．
3. 選択したい項目が見つかり，その上にマウスが止まったとき，「これ！」という指示のつもりでボタンクリックなどをする．
4. クリックされたオブジェクトに対して，あらかじめ関連づけられた機能が呼び出される．(領域に対してではない．)

図 2 に Presentation を使ったらどういうことまでできるかを示してみた．適当な表示をし，その表示をする際に，これにはこういう手続きと関係があるということを示しておく．そうすると，その表示をするのに使われた領域は自動的に理解されているので，その領域の上をマウスが通り，それをクリックすると対応する処理が起動されるというようにできる．

4.2 Presentation

Presentation は簡単にいえば，「表示されたものもまだ生きている」ということである．Presentation 機能を使って画面に表示されたものは，その型，各種の属性が同時に記憶されており，それによって後に生じる再入力に備えられる．どれだけのかたまりを一つのオブジェクトとするかは表示時に制御できる．

型（正確には presentation type）としては通常の型，クラスに加えて，新しい型を定義できる．それらの間には半順序関係が規定される．

基本的には通常の print に相当する present プリミティブで表示し，read に相当する accept プリミテ

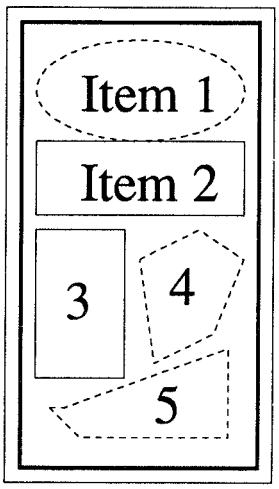


図 2 Presentation によるメニュー(の極端な例)

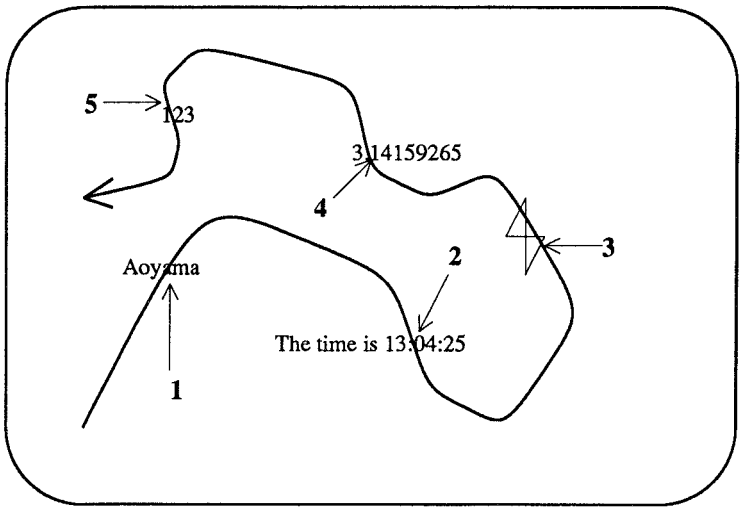


図 3　Presentation 上でのマウスが通りすぎる過程

ィブで再入力するという使い方をする．accept 指示ではほしいオブジェクトの型を指定し，実行されるとその型（もしくはその副型）のオブジェクトだけが再入力対象となる（文脈依存マウスセンシティビティといわれる）．それらをマウスでセンスすることだけで入力できる．（もちろん，そこでキーボードから新たに打ち込んでもよい．）また，さらに高位のインタフェースも用意される．これらはもともと Lisp マシンで育てられた機能である．詳細な最新の仕様／概念を調べるには Symbolics のマニュアルを見るのが最も適している．

たとえば，`(accept '((integer) :description "the number of copies"))` とすると

```
Enter the number of copies:
```

と表示され，マウスセンス入力かキーボード入力が待たれる．マウスセンスの対象には integer として present されたものがなる．通常の print 構文での出力は適切な型を伴った present になるように自動的にされているウィンドウがあるとすれば，そこへのすべての表示はマウスセンスの候補となる．（もちろん，そうしなくともよい．）

「Presentation 型指定子」という概念があり，これにより，通常の型指定子のように型の and, not, or をつけたり，member をしたり，あるいは範囲を指定したりできる．こうした機能は，Symbolics 以外では CLIM, Express Window, Y_Y などで備えられている．また，Apple 社の pink での実験でもこうした機能が試されているようである．

さまざまな default の指示，あるいは多段の選択がある場合に上位レベルでの選択により下位のメニュー内容を変えるとか，さらにカストマイズした動作を規定できる．

図 3 は，present されたものの上を accept の文脈中にマウスが通りすぎる過程を表わしている．その文脈で対象にできる型のオブジェクトがそこに表示されている場合，それらはハイライトされる．

興味深い使い方をいくつか紹介する．

図中，3 のような複合的な図形を一つのオブジェクトとして表示し，それに対するクリックが数値を返すとか，ある動作を引き起こすとかいった仕組みにすることもできる．また，Aoyama という表示があるが，これを fixnum 型のオブジェクトとしておき，その文脈で Aoyama をクリックするとまさに fixnum の数値

が返されるようにすることもできる.

これらのことを容易にできるようにするマクロなどが，もともとの仕様である Symbolics に準拠して用意されるが，それらは多少の違いがある.

5. 前提となる処理系の機能について

今日，多数の Lisp 処理系が存在するが，いったいどのようなものがあるのか，それらのうちどのようなものが，ここで述べているようなウィンドウツールキットをもち得るのか？ 読者の興味もあると思われるので，それに関連する表を前述の電子協の調査資料から示す.（なお，これらの調査結果が完璧に日本での利用状況を反映しているとはいい切れない.）

表 2 での単位は%，複数回答あり. なお，Sun CL と Lucid CL は実体としては同じであるが，もともとの集計を尊重してそのままにしてある.（なお UtiLisp, Franz Lisp, Scheme などは Common Lisp ではない.）

3節の冒頭で述べた分類の中で，1）に「基本的には何もない」ということに触れたが，本章で対象とするような「基本的には何もない」実用的処理系とオモチャの処理系との間には，大人と子供ほどの違い，あるいは近代的高層建築と掘建て小屋ほどの違いがある.「基本的には何もない」というのは，ある意味では何色にも染めていないだけであるといえる. たとえば現在，UNIX 上で利用できる Allegro, Lucid, CMU Python, KCl などの処理系それ自身はこれである.

ウィンドウツールキットが動かせる処理系はどうなっていればいいか？ このことに触れておきたい.

まず第一に，何らかの仕組みで OS の諸機能との会話ができなければならない. 非同期に生じるイベントの処理ができなければならない. これを実現するには，割込み処理機構もしくはマルチプロセス機能が必要と

表 2 電子協でのアンケート集計結果（処理系）より

——1985——	——1987——	——1990——
Franz Lisp (44.4)	KCl (32.8)	KCl (21.6)
Vax Lisp (29.6)	Franz Lisp (20.3)	UtiLisp (15.8)
InterLisp-D (24.4)	Common Lisp (18.8)	Sun CL (13.5)
UtiLisp (24.4)	Vax Lisp (18.0)	Allegro (11.3)
		Emacs (11.3)
		GCLisp (11.3)
		Symbolics (11.3)
Zetalisp (13.3)	UtiLisp (8.6)	Xerox Lisp (7.7)
KCl (12.6)	Interlisp-D (8.6)	Lucid CL (7.2)
—	GCLisp (8.6)	Elis CL (6.3)
		Franz Lisp (6.3)
—	Zetalisp (6.3)	Interlisp-D (5.4)
		Vax Lisp (5.4)
—	TAO (5.5)	ICLisp (4.1)
		VOS 3 Lisp (4.1)
—	VOS 3 Lisp (5.5)	XLisp (3.2)
		Scheme (3.2)
135 replies	128 replies	222 replies

なる．

次に，Lisp にはガベージコレクタがつきものであるので，これとの相克の問題が生じる．

ガベージコレクタが，こうしたシステムインタラクション関連の中間状態を誤って回収しようとしてしまったり，ガベージコレクタの動作中にも待ってくれないイベントへの対応を忘れたりすることが考えられる．こうしたことへの対処がちゃんとしていなければならない．また，ガベージコレクタの作動時間を極小化する，あるいはリアルタイム化する展開も必要になってくる．

これらのことがOKであれば，その処理系はおそらく GUI をインストールできる能力ができたといえる．もちろん，それに相応した各種の機能の豊富さも平行して重要である．

6. いくつかのツールの紹介

1991年10月30日から11月1日まで，Washington DC の近くの Gaithersburg で開かれた Lisp Users and Vendors Conference '91（LUV '91）では，User Interface Design Tools Panel があった．そこでまとめられたツールは Action!, Allegro Common Windows, CLIM, CLM-Gina, ezd, Garnet, LispView, Lisp Works, Medley, Picasso, Winterp, *YyonX* であり，そのなかで CLM-Gina, Picasso, Winterp, CLIM, Garnet の発表があった．以下に，その時点でまとめた情報を示す．また，このときには紹介されなかったものの中から CLIO を最後につけておく．

6.1 Action!

・ベースとなるウィンドウシステム：Macintosh, OS/2 PM, Motif+X Windows
・利用できる Lisp 処理系：TI Common Lisp, Macintosh CL, Procyon CL, Sun（Motif）CL
・開発者：ExperTelligence Inc.
・種別：商品
・ツールのタイプ：Interface+Object Builder
・既存ユーザインタフェースライブラリとのインタフェース：Macintosh ToolBox, X, PM, Motif widgets
・特徴的な機能：interactive, open. Action! のオブジェクトはすべてユーザが修正・生成・表示できる
・概要：Action! は intelligent interactive, object-oriented なグラフィックユーザインタフェースかつオブジェクト開発システムである．オブジェクト指向である．そもそもは Macintosh 用

6.2 Allegro Common Windows

・ベースとなるウィンドウシステム：X 11
・利用できる Lisp 処理系：Allegro Common Lisp ver. 3.1 or 4.1
・開発者：Franz Inc.
・種別：商品
・ツールのタイプ：GUI ツールキット
・既存ユーザインタフェースライブラリとのインタフェース：なし（X のライブラリとのインタフェースあ

り)

・特徴的な機能：高性能，汎用のウィンドウシステム

・概要：CLOS ベース．Common Windows の仕様を X 上で実現し，さらに機能拡張した

6.3 CLIM

・ベースとなるウィンドウシステム：Symbolics Genera, X 11, Macintosh Toolbox, Windows 3.0

・利用できる Lisp 処理系：Symbolics Genera, CLOE on DOS, Franz Allegro, Lucid CL, Apple MCL 2.0, Harlequin Lisp Works

・開発者：International Lisp Associates

・種別：商品

・ツールのタイプ：基本的には UIMS である．ポータブルツールキットのようにも扱える

・既存ユーザインタフェースライブラリとのインタフェース：CLM を介して Motif, LispView を介して OPENLOOK. 統合的な UIMS としてケースバイケースでカストマイズできる

・特徴的な機能：さまざまなプラットフォーム上でのポータビリティ．presentation type を介したアプリケーションセマンティクスとの結合．層化された構造により拡張性が高い．CLOS ベース

・概要：Common Lisp Interface Manager の頭文字に由来する．そもそもは Symbolics Genera での機能を他のプラットフォームで実現するためのもの

6.4 CLM-GINA

・ベースとなるウィンドウシステム：X 11

・利用できる Lisp 処理系：Franz Allegro, Lucid, Genera, その他 CLX と CLOS をもつもの

・開発者：Project GINA, German National Research Center for Computer Science

・種別：研究．無料入手可能

・ツールのタイプ：CLM は Motif toolkit への Lisp インタフェース．GINA は UIMS（フレームライブラリ）でインタフェースビルダをもつ

・既存ユーザインタフェースライブラリとのインタフェース：Motif

・特徴的な機能：Motif, CLOS, CLX の上の強力なハイレベルレイヤを提供．よく用いられるようなダイアログのためのコードは用意されている

・概要：CLM は Common Lisp Motif interface に，GINA は Generic Interactive Application にそれぞれ由来する．GINA は CLM を介して Motif ツールキットを用いるアプリケーションフレームワーク．GINA のクラス群のサブクラスとして応用を記述することで機能を利用．インタフェースビルダがあり，これにより，ウィンドウレイアウトの設計と Motif オブジェクトのパラメータが与えられる．IB は GINA を利用する Lisp コードを生成する

6.5 ezd

・ベースとなるウィンドウシステム：X 11

・利用できる Lisp 処理系：Scheme ベース．基本的には DECStation 用
・開発者：DEC
・種別：研究．'91 年 6 月に出された研究レポートがある．Bartlett, Joel F.：Don't Fidget with Widgets, Draw!, WRL Research Report 91/6, May 1991
・ツールのタイプ：グラフィックサーバ
・既存ユーザインタフェースライブラリとのインタフェース：なし
・特徴的な機能：グラフィックサーバとして，アプリケーションプログラムと X サーバとの間に位置し，ストリームを介して相互の通信を行なう．アプリケーションがグラフィックオブジェクトを定義するとそれが描画指示として扱われる．それは ezd により，ウィンドウにマッピングされる
・概要：ezd は easy drawing for programs on X displays に由来する

6.6 Garnet

・ベースとなるウィンドウシステム：X 11
・利用できる Lisp 処理系：Lucid, Allegro, Sun, Harlequin, TI, Symbolics, AKCL
・開発者：Carnegie Mellon University
・種別：研究．USA, Canada, UK, Australia ではライセンス取得により無償利用可能．*IEEE Computer*, Nov., 1990, Vol. 23, No. 11, pp. 71-85 に資料がある
・ツールのタイプ：Tookit, UIMS and Interface Builder
・既存ユーザインタフェースライブラリとのインタフェース：なし．しかし Motif-like な look and feel を Lisp でエミュレート
・特徴的な機能：あらゆるユーザインタフェース機能をカバーすることを考えている
・概要：Garnet は Generating an Amalgam of Real-time, Novel Editors and Toolkits に由来する．X 11, Xlib からは独立している．表示の自動管理，データの自動レイアウト，組込みの高レベル入力イベント処理などがある．2 種類のウィジェットセットをもつ（うち一つは Motif-like)．また，Graphical constraints という概念がある．プリントのための自動 PostScript 生成機能がある．Gilt と呼ぶインタフェースビルダがある

6.7 LispView

・ベースとなるウィンドウシステム：X 11
・利用できる Lisp 処理系：SunCL
・開発者：Lucid/Sun
・種別：商品であった．Sun Common Lisp 4.0 に付属（なお 1991 年 11 月に，ソースコードが公開された．）
・ツールのタイプ：Toolkit（Devguide インタフェースビルダ対応のコードジェネレータがある）
・既存ユーザインタフェースライブラリとのインタフェース：X lib and X View
・特徴的な機能：CLOS ベース
・概要：開発当初の段階では SOLO と呼ばれた．Open Windows Developer's Guide（Devguide）ファイルを Lisp View に翻訳する機能がある．（Devguide は Sun から提供される UI ビルディングツール．）

OPEN LOOK GUI スタイルを実現．(value obj) とするだけでスライダの値を取ったりできる．

また，LispView 1.1 は X の contrib に入り，MIT から ftp できるようにもなった (`ftp.csrl.aoyama.ac.jp` にもある)

6.8 LispWorks

・ベースとなるウィンドウシステム：X 11 (CLUE を使用)
・利用できる Lisp 処理系：Harlequin CL
・開発者：Harlequin Ltd.
・種別：商品
・ツールのタイプ：Toolkit, Interface Builder あり (Motif, OPEN LOOK, CLIM スタイルをサポート)
・既存ユーザインタフェースライブラリとのインタフェース：Xlib
・特徴的な機能：Lisp Editor, SQL interface, hypermedia authoring system, Lisp-based Prolog などが具備されている．この上に Knowlegde Works (知識ベースシステム開発のためのツールキット) や Data Works (リレーショナルデータベースへの知的インタフェース開発のためのツールキット) などが別途用意されている
・概要：2 年間の使用実績がある．**図 4** に画面図を示す

6.9 Medley

・ベースとなるウィンドウシステム：X 11
・利用できる Lisp 処理系：Medley
・開発者：Venue Corp. (もともとは Xerox)
・種別：商品
・ツールのタイプ：フルサービスの Lisp 環境
・既存ユーザインタフェースライブラリとのインタフェース：なし．Xerox の Interlisp 系のユーザツールを受け継ぐ
・特徴的な機能：スペース効率がよい．多数のユーザ・ベンダ開発のツールをもつ成熟したシステム．Rooms (多重仮想デスクトップ) がある．LOOPS が使える
・概要：Medley: そもそもは Xerox Lisp の最新版に対するコードネーム．386, SPARC, IBM, DEC のワークステーション上に移植され，Xerox 1100 系のすべての機能がある

6.10 Picasso

・ベースとなるウィンドウシステム：X 11
・利用できる Lisp 処理系：Allegro CL
・開発者：Lawrence A. Rowe, Univ. California Barkeley
・種別：研究．無料入手可能
・ツールのタイプ：ツールキットとアプリケーションフレームワーク

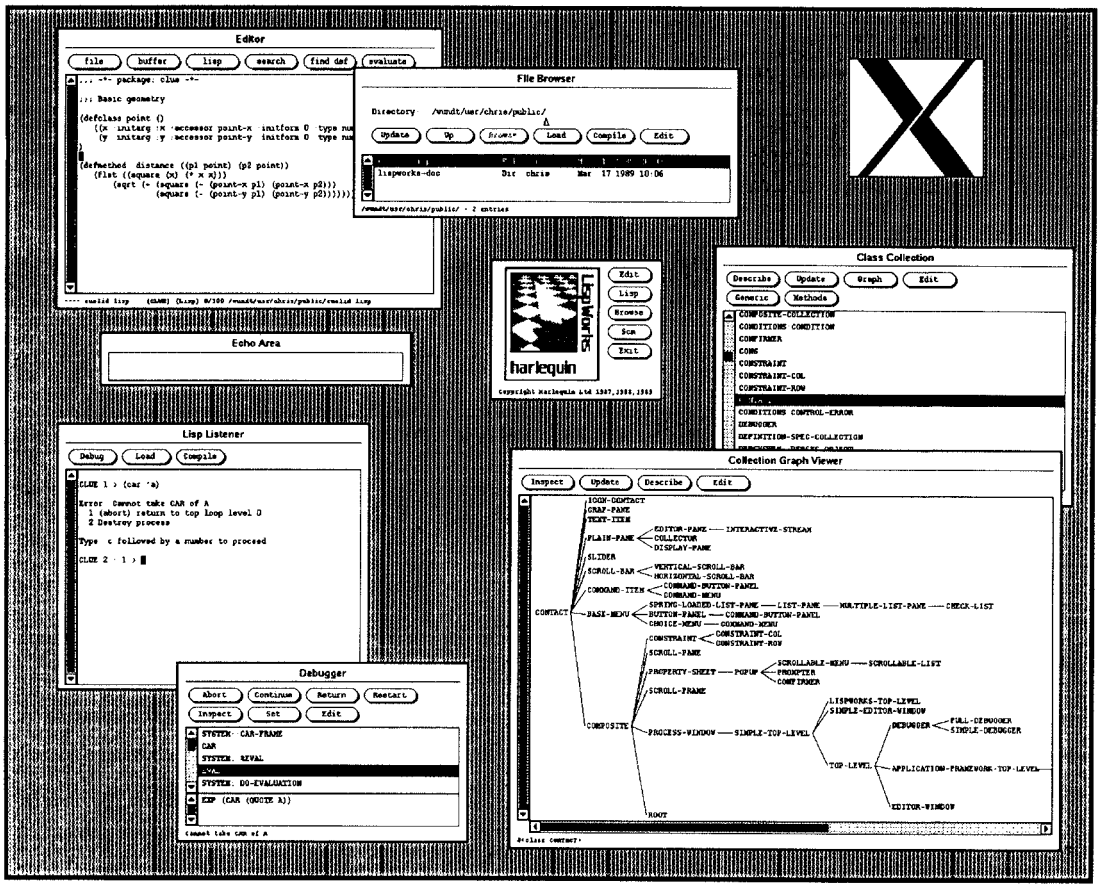


図 4　Lisp Works の画面例

- 既存ユーザインタフェースライブラリとのインタフェース：なし
- 特徴的な機能：多数の高度なウィジェット（graphics, video, tables）が用意されている．強力なアプリケーションフレームワーク（再使用可能ダイアログ，パネルパラメータ）をもつ．multiple geometry managers や data propagation 機能をもつ
- 概要：Picasso は GUI 開発システムである．上に述べた機能のほかに constraint システムもある．CLOS ベースである

6.11 Winterp

- ベースとなるウィンドウシステム：X 11
- 利用できる Lisp 処理系：XLisp（David Betz による無料の Common Lisp サブセットインタプリタ smaltalk-like のオブジェクト指向機能がある）
- 開発者：Human-Computer Interaction Dept. Hewlett Packard Labs.
- 種別：研究．無料入手可能
- ツールのタイプ：X ツールキット Xt と Motif ウィジェットへのハイレベルインタフェース
- 既存ユーザインタフェースライブラリとのインタフェース：なし

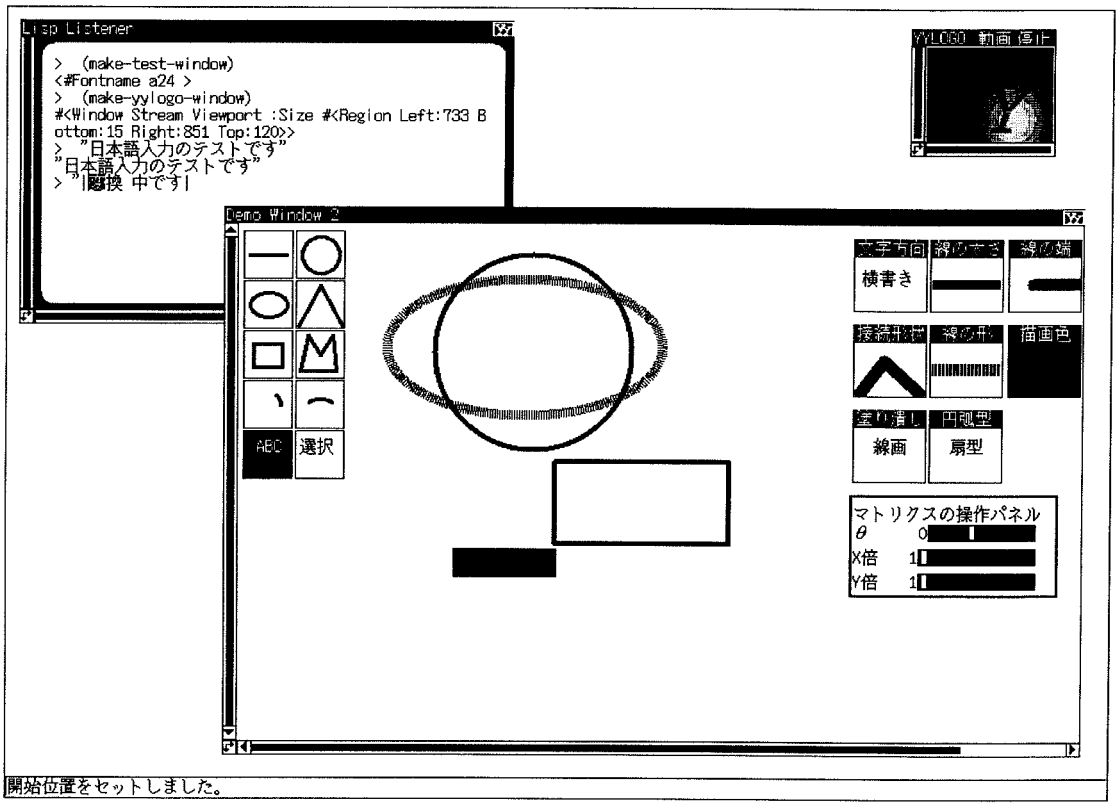


図 5 *YyonX* の画面例

図は，三つのサブウィンドウからなる *YY* の画面例である．これ全体が X ウィンドウ下の一つのウィンドウとなっている．各サブウィンドウの管理は *YY* サーバが，内容表示は *YY* クライアントが行なっている．この例では，*YY* クライアントは ELC，X サーバおよび *YY* サーバは同一イーサネット上の Sun 4-260 を用いた分散協調動作を示している．*YY* ロゴはカラーアニメーションウィンドウの例ともなっており，実際に見てもらえるとわかるが，動く．お絵書きのウィンドウでは描きたいもの，その種類をマウスで選択し，作画できる．

・特徴的な機能：（OSF/Motif の）ウィジェットは XLisp のオブジェクト機能のオブジェクトである．会話型の X/GUI プログラミング Gnu-like な構造（小さくて高速な Lisp インタプリタと C による実現）

・概要：Winterp は The OSF/Motif Widget INTERPreter の意味である．OSF/Motif の UIL などの言語を使うより柔軟性があり，カストマイズ／プロトタイプによい

6.12 *YyonX*

・ベースとなるウィンドウシステム：X 11

・利用できる Lisp 処理系：Symbolics, Lucid, Allegro

・開発者：青山学院大学

・種別：研究．無料入手可能

・ツールのタイプ：ツールキット

・既存ユーザインタフェースライブラリとのインタフェース：なし

・特徴的な機能：分散環境指向．Lisp によるアプリケーションサーバ（*YY* クライアント）とネットワークでつながれた *YY* サーバがリアルタイムに会話し，それを別の X サーバを使用する利用者が利用できる．負荷分散と遠隔実行を実現
・概要：*YYonX* はネットワーク上の分散ユーザインタフェースを研究する *YY* プロジェクトの X ウィンドウ上での実現である．**図 5** に例を示す

6.13 CLIO

・ベースとなるウィンドウシステム：X 11
・開発者：TI．X の配布に含まれている
・種別：実験研究
・ツールのタイプ：CLUE 上のウィジェットセット
・既存ユーザインタフェースライブラリとのインタフェース：CLUE（Xt レベル）
・特徴的な機能：CLUE でいう contacts としての機能充実が中心となっている
・概要：特定の look and feel の規定はせず，既存の look and feel を真似できるようにしてある．現在は OPEN LOOK 仕様の機能が外側に用意されている

7. あとがき

X ウィンドウ上で使える Common Lisp 処理系，そしてその上でのツールについて紹介した．あわせて，まさに Lisp 的概念である Presentation について簡単に紹介した．商用の処理系もいくつか紹介したが，時代は刻々と移り変わるものであり，また，著者から見た編集上のバランスの点から省かれているものもある．その結果，ここに記したものだけがすべてではないことを最後にお断わりしておく．

（いだ まさゆき　青山学院大学 情報科学研究センター）

目　次

石田 晴久 編

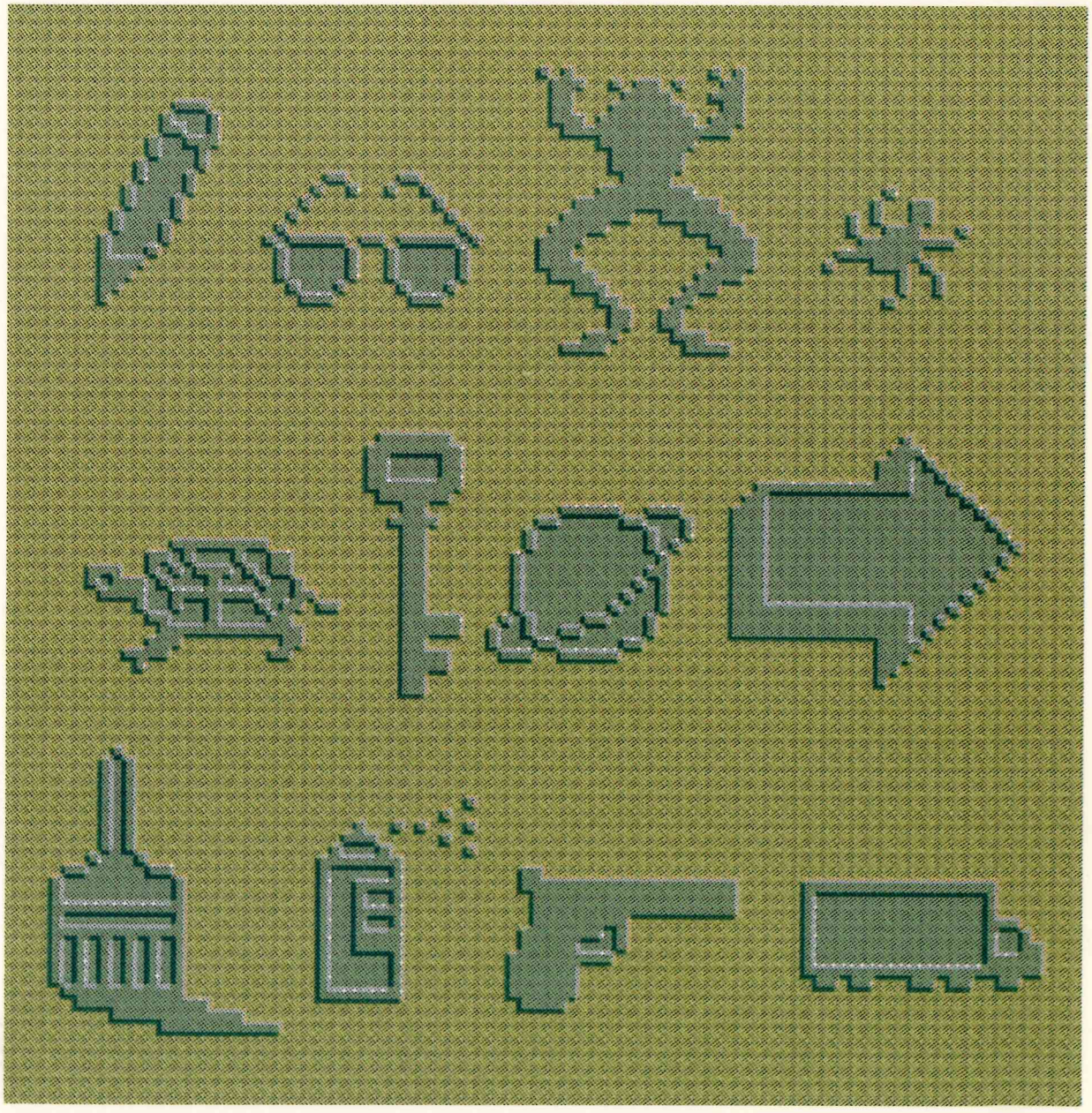
Xウィンドウ

とその仲間たち

コンピュータ・サイエンス誌 bit

別冊

共立出版



C&C Computers and Communications

NEC

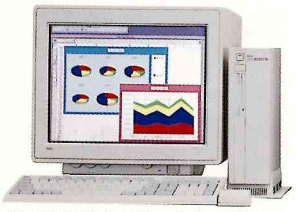
第2のビジネス革命もNEC。

PC98でビジネスを変えたNECが、日本のビジネスをまた変える。
第2のビジネス革命を起こすNECのUNIXワークステーション、EWS4800シリーズ。

情報システムの革命が起きる。これからのビジネス戦略の中核となるクライアント／サーバ コンピューティングがさまざまなビジネスシーンにやって来る。**グラフィックスの革命が起きる。**1280×1024ドットの高解像度画像と※1670万色を同時表示するグラフィックスパワー、そして描画スピードの速さが、3次元グラフィックス、マルチウィンドウをらくらく実現。**スピードの革命が起きる。**メインフレームなみの28 MIPS～37 MIPSの高速処理性能がマルチタスクに威力を発揮。**ネットワークの革命が起きる。**オープンシステムにより、世界のさまざまなコンピュータとリアルタイムにネットワーク。

◎あなたの業界でも、第2のビジネス革命が始まる。

※対象機種はEWS 4800／230、／260、／75。



RISC搭載世界最小のワークステーション。

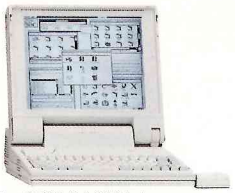
EWS 4800/210

性能／28MIPS　諸元／32ビットCPU（RISC）V R3600（25MHz）搭載
●メインメモリ：16～48Mバイト　●最大ディスク容量：3.7Gバイト
●内蔵ディスク容量：418Mバイト　●3.5インチFDD：1台
●外形寸法：90×270×290（W×D×H mm）

サーバとしても活躍するRISCスタンダード・モデル。

EWS 4800/230

性能／37MIPS　諸元／32ビットCPU（RISC）V R3600（33MHz）搭載
●メインメモリ：32～128Mバイト　●最大ディスク容量：7.28Gバイト
●内蔵ディスク容量：418/662Mバイト　●3.5インチFDD：1台
●外形寸法：430×405×150（W×D×H mm）



数々のクラス最高性能を搭載したラップトップ・モデル。

EWS 4800/120LT

性能／28MIPS　諸元／13インチTFTモノクロ液晶画面
●32ビットCPU（RISC）V R3600（25MHz）搭載　●メインメモリ：16～48Mバイト
●最大ディスク容量：3.7Gバイト　●内蔵ディスク容量：418Mバイト
●3.5インチFDD：1台　●外形寸法：340×399×116.5（W×D×H mm）

画面はハメコミ合成です。

《いよいよ、  新ビジネス体制が来る。》

古舘伊知郎

OPEN SYSTEM SUPER STATION

NECスーパーステーション
EWS 4800シリーズ

X Open UNIX 1170

EWS-UX/V(Rel 4.0)は、日本で初めてオープンシステムを推進する国際標準化団体X/Openから、公認プラント（XPG3）を獲得しました。

＊UNIXオペレーティングシステムは、UNIX System Laboratories, INC.が開発し、ライセンスしています。
＊X/Openおよびマークは、X/Open Company Limitedの商標です。

NECスーパーステーションEWS4800シリーズ
NECスーパーサーバーUP4800シリーズ

EWS 4800/15　EWS 4800/35　EWS 4800/75　EWS 4800/120LT　EWS 4800/210　EWS 4800/230　EWS 4800/260　UP 4800/520

日本電気株式会社

製品に関するお問い合わせは、ワークステーション事業部へ　℡(03)3798-6288 東京都港区芝5丁目7-1 〒108-01
資料のご請求は、右の資料請求券を官製ハガキに添付し、住所・氏名・年令・電話番号・会社名・所属をご記入の上、上記宛先までお申し込みください。

資料請求券 EWS bit別冊