Apple Computer, Inc.
The Lisa Applications Tool Kit

1983

What is a "Lisa-style" application?

Some of the hallmarks of a "Lisa-style" application are:

* desktop icons
* overlapping windows
* cut-and-paste between windows
* pull-down menus
* scroll bars
* dialog boxes
* alert boxes

The user interface is "see and point" instead of "remember and type."

Can a Lisa-style application run under Xenix or CP/M?

No. Lisa-style applications only run under Apple's Proprietary Operating System.

Can a Lisa-style application be written in Basic or Cobol?

No. At present, Lisa-style applications must be written in Lisa Pascal. MC68000 machine-language subroutines may be written as well. You can run Basic and Cobol programs in the Lisa workshop environment but they will not look like Lisa-style applications.

How do I edit, compile, and debug a Lisa-style application?

Apple's operating system has several command shells, including:

* The Desktop Manager
* The Workshop Shell

At first release of the Tool Kit, applications will be edited and compiled under the Workshop Shell and will be executed under the Desktop Manager. Before testing your application, you will have to switch shells, which takes a few keystrokes and about one minute.

The Workshop editor is itself Lisa-style (e.g., you can use the mouse and menus), except that you type file names instead of pointing at icons, and there are no other applications on the screen at the same time. The rest of the Workshop user interface is reminiscent of UCSD Pascal and, hence, does not use the mouse or windows.

In a later release of the Tool Kit, it may be possible to edit and compile under the Desktop Manager and avoid switching shells.

1/15

I want to port an application from another computer to the Lisa in a hurry. Should I use the Tool Kit?

If you want to port a conventional "remember and type" or "prompt and type" Pascal program to the Lisa, you should not use the Tool Kit. However, you can port these so-called "vanilla" applications to the workshop environment and run them there. If your program is written in Pascal or Assembler, it can call the QuickDraw graphics package as well as obtain input from the keyboard and mouse.

If you want to have your vanilla application eventually run in a window on the Desktop, you should use Pascal and the Workshop now, and the conversion will be easier later.

If I port a "vanilla" application to the Pascal Workshop, what can I do to make it easier to move to the Tool Kit later?

Modularize your program to clearly isolate:

* I/O (User input, Display management, Filing)
* Data structures and operations on those structures

When you move to the Tool Kit, your I/O will change substantially, but your data structures and computations will not have to change as much.

Lisa-style applications have an event-driven control structure, as follows:

Repeat:

* Get an event from the user (for example, the user selects a menu item).
* Call an appropriate procedure for each type of event
* Update the display accordingly

Figure 1 illustrates this type of program control. The closer your program is to this organization, the easier it will be to convert it to a Lisa-style application. This summer, we will provide you with more detailed technical information on how to structure your programs to simplify later conversion to the Tool Kit.

Will it be easy to write Lisa-style applications using the Tool Kit?

Yes and no. It is easy in that most of the Lisa-style aspects are implemented automatically. It is hard in that the control structure and programming style are unusual and will take some getting used to. However, once you get used to the Tool Kit you will find it extremely powerful. We are working on a comprehensive set of training and reference materials for the Tool Kit in order to get you up to speed as soon as possible.

How are most Lisa-style aspects of the application implemented automatically?

Lisa applications have to deal with a myriad of details to conform with the Uaer Interface Standard, initialize properly, terminate properly, coexist with other applications, cut and paste with other applications, run in overlapping

2/15

windows, file through desktop icons, expand and contract the working set as needed, recover from crashes, print on different printers in multiple fonts, and so on.

Rather than have all this code duplicated and debugged in every application, the Tool Kit includes a Generic Application that implements all standard Lisa application behavior. All your have to do is say how your application differs from the standard.

That means you have to define application-specific data structures, algorithms, and visual presentations, but you do not have to write code to deal with windows, icons, scrolling, resizing, menus, and so on.

The Generic Application implements the event-driven control structure of a Lisa-style application. It can handle some events entirely on its own, such as filing, printing, scrolling, resizing, and splitting a window into views. When it comes time to display the document, highlight the selection, handle a keystroke, or respond to a press of the mouse button over the document, the Generic Application calls procedures that you provide to implement application-specific behavior.

All but a few lines of the Generic Application are precompiled and prelinked into a library that your application uses (in the UCSD Pascal sense).

In UCSD Pascal, if my application uses a library, I can call its procedures, but it can't call mine. Is the same true in Lisa Pascal? If so, how can the Generic Application make calls on my application-specific procedures.

As in UCSD Pascal, a library cannot call ordinary procedures in an application. However, Lisa Pascal has been extended to include a new kind of procedure called a method that can be defined in an application and called by a library. Methods are associated with classes, which provide a way to define generic behavior in a library which can be overidden in an application should a developer want a modified behavior.

Classes and methods are among a set of extensions to Lisa Pascal that are called the Clascal extensions.

What is Clascal?

Clascal adds a new type to Pascal called OBJECT, a new type declarator called SUBCLASS, and other new constructs. Clascal is not a completely new concept. Simula-67 added classes to Algol-60 sixteen years ago; Clascal simply does the same for Pascal today.

Clascal is easier to learn than either Simula-67 or Smalltalk-80 because its syntax is Pascal-like, and all Pascal features are available. However, if you become familiar with Simula-67 or Smalltalk-80, you will have an easier time learning to use Clascal.

How can I learn about Smalltalk-80?

The following references would be helpful:

3/15

* Smalltalk-80: The Language and its Implementation. Adele Goldberg and David Robson, Addison Wesley: 1983

* Byte's special issue on SmallTalk. August, 1981.

At times, you may find this material heavy-going. Don't dispair, Clascal is much easier to learn than Smalltalk. Your objective when reading these articles should be to understand the fundamental concepts of object-oriented programming (classes, objects, methods, sub-classes etc..) rather than trying to learn SmallTalk, per-ce. Larry Tesler's article in the Byte issue is very good for understanding the design philosophy which underlies much of our software.

Whats the relationship between Clascal and the Tool Kit?

The Tool Kit was written using the Clascal extensions, and in order to use the Tool Kit you will have to understand and use Clascal. However, only those portions of your application which communicate with the Tool Kit have to be written using Clascal.

How do I do text editing and put up dialog boxes?

These and other standard functions are available through Building Blocks that your application may incorporate. Among the Building Blocks Apple plans to produce are:

* Text editing (as in LisaWrite)
* Structured graphics (as in LisaDraw)
* Plotting (as in LisaGraph)
* Bit-map graphics (icons and fonts)
* Data base access (as in LisaList)
* Field entry checking and output formatting (as in LisaList)
* On-line instruction (as in LisaGuide) and on-line reference (Help)
* Networking and Data communication
* Dialog Boxes

How does an applications designer decide when to use menus and when to use dialog boxes?

These and other user interface design questions will be answered in the Lisa User Interface Standards document, a component of the Tool Kit.

How do I debug a Lisa-style application?

The Tool Kit includes the same low-level debugger as the Workshop. In addition, there are numerous Tool Kit-specific high-level symbolic and visual debugging aids.

4/15

How do I tune the performance of a Lisa-style application?

The Tool Kit includes a number of performance measurement tools, and the Tool Kit Guide suggests ways to improve performance. Tool Kit applications will generally perform even better than the applications Apple has already demonstrated because the Tool Kit takes advantage of more recent enhancements to the Lisa applications architecture.

When will the Workshop and the Tool Kit be available?

The Workshop is scheduled for mid-1983. The Tool Kit will be released in stages. At the end of 1983 we plan to release:

* A Generic Application
* A Text Building Block
* A Dialog Box Building Block
* The Lisa User Interface Standard
* The Tool Kit Reference Manual
* The Tool Kit Tutorial
* A Training and Support Package
* Debugging and Performance Measurement Aids

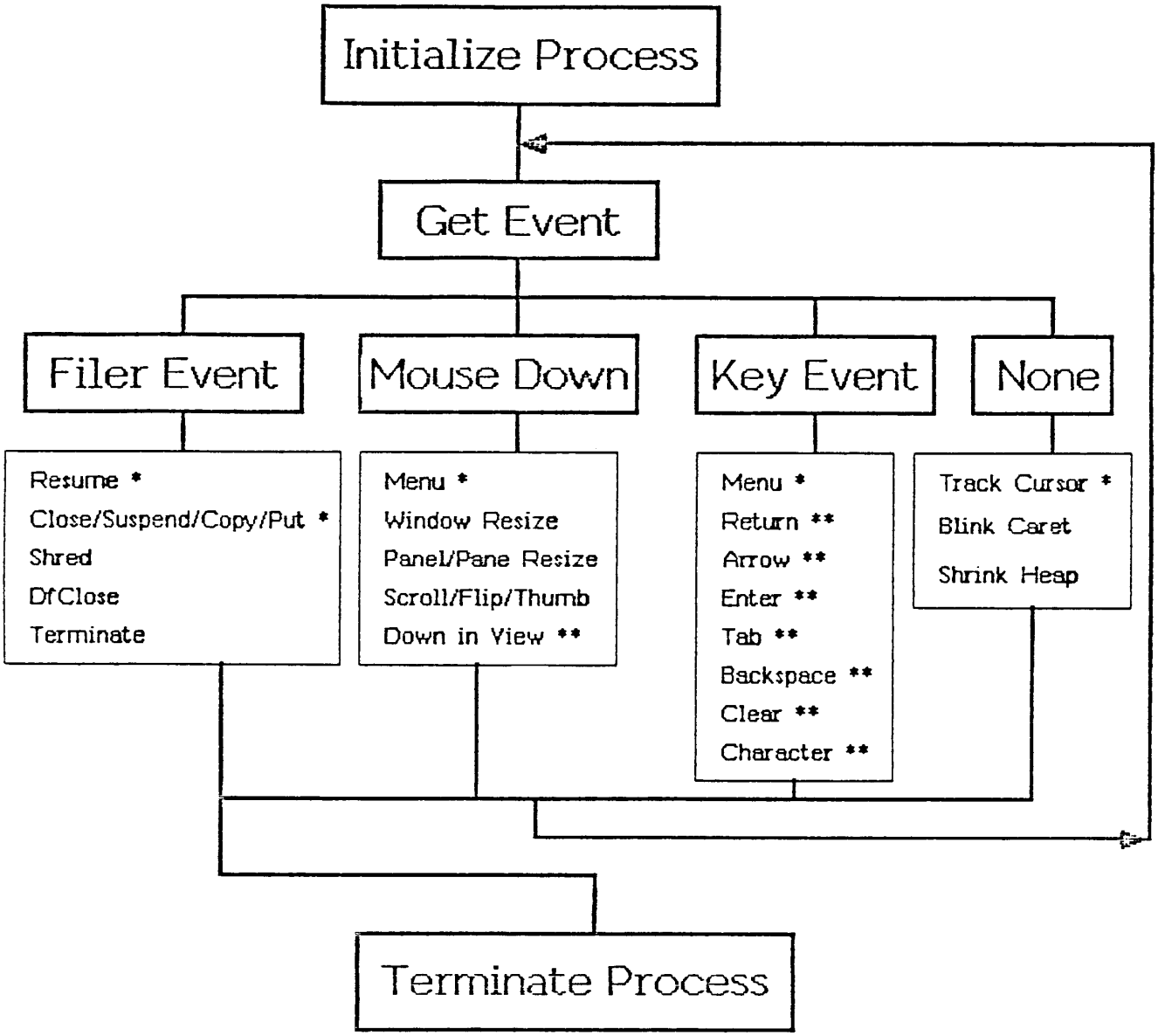
Other Building Blocks are planned for 1984.

If Apple systems software changes, will my customers still be able to run the applications they bought from me?

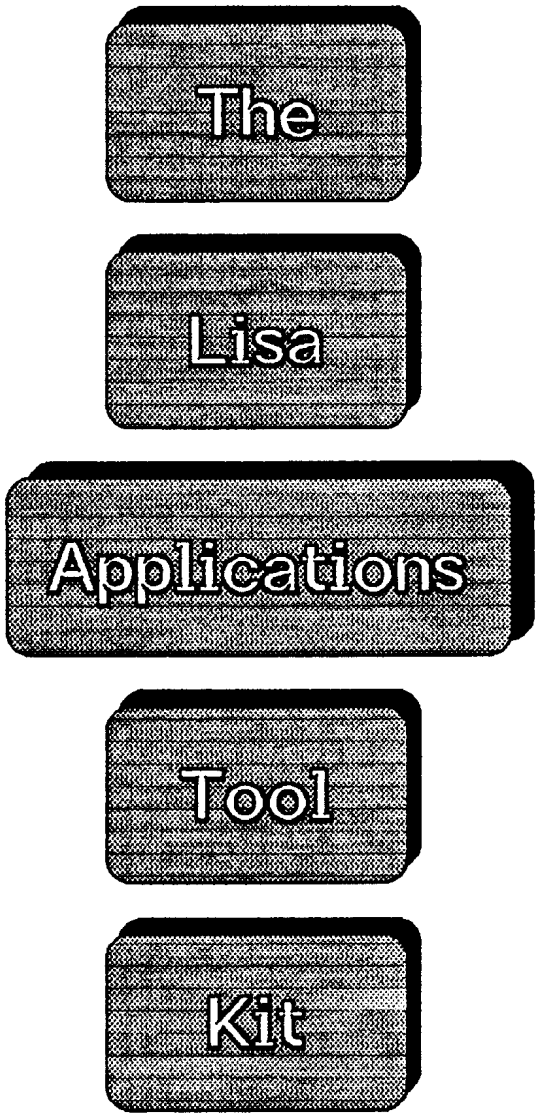
In 1983, it may be necessary to recompile and relink applications in the event that Apple makes certain kinds of changes to the systems software. Starting some time in 1984, Apple will provide a "stable and extensible" base of systems software such that applications that use it will not have to recompile or relink any more.

5/15

Tool Kit Control Flow

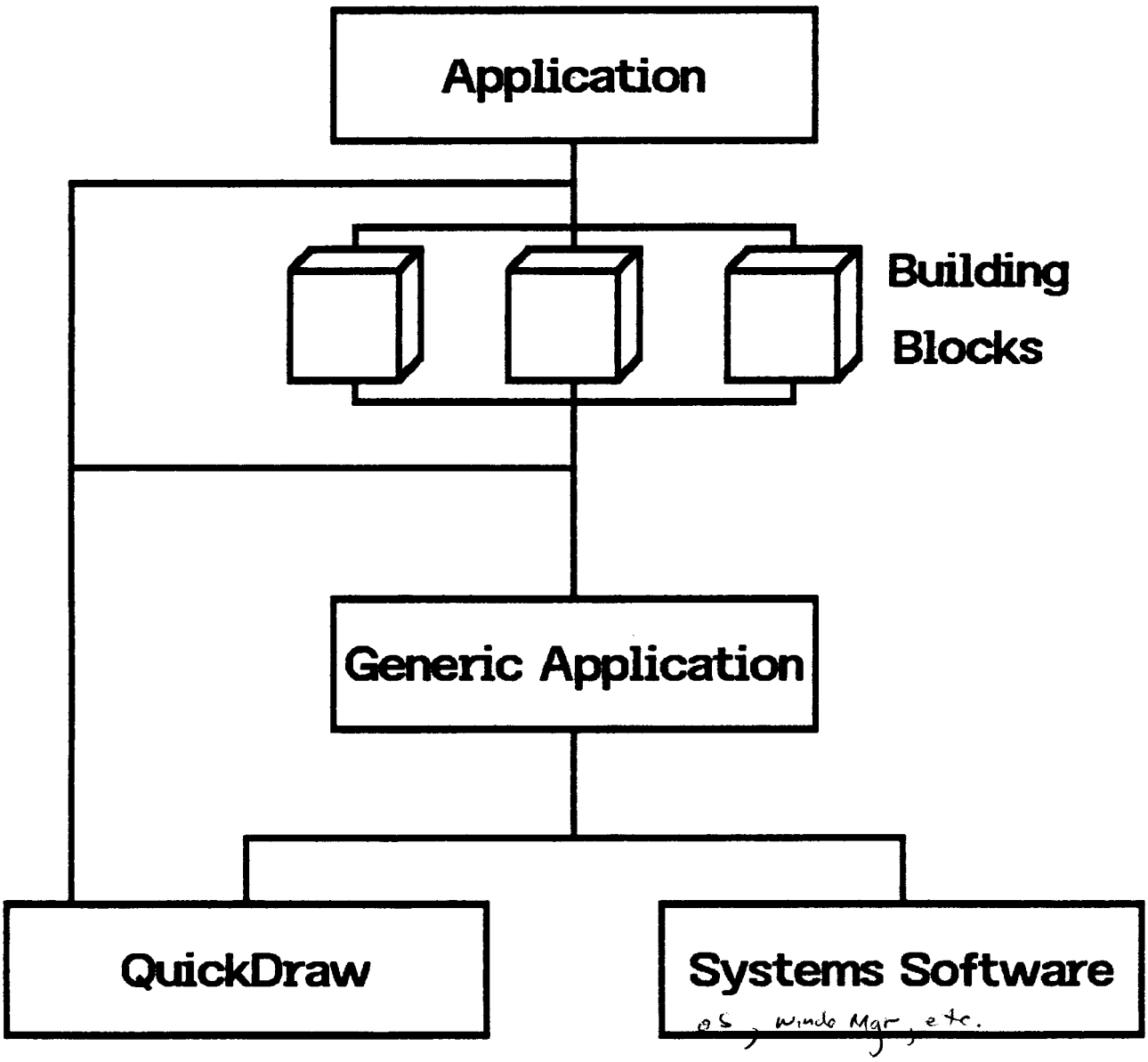


6/15



7/15

Lisa Applications Software Hierarchy



8/15

Tool Kit

Building Blocks

Paragraph Editing (ala LisaWrite) 1983

Structured Graphics (ala LisaDraw)

Bit Map Graphics (fonts and icons)

Network & Telecommunication

Tutor (ala LisaGuide) 1984

International Data Entry and Formatting

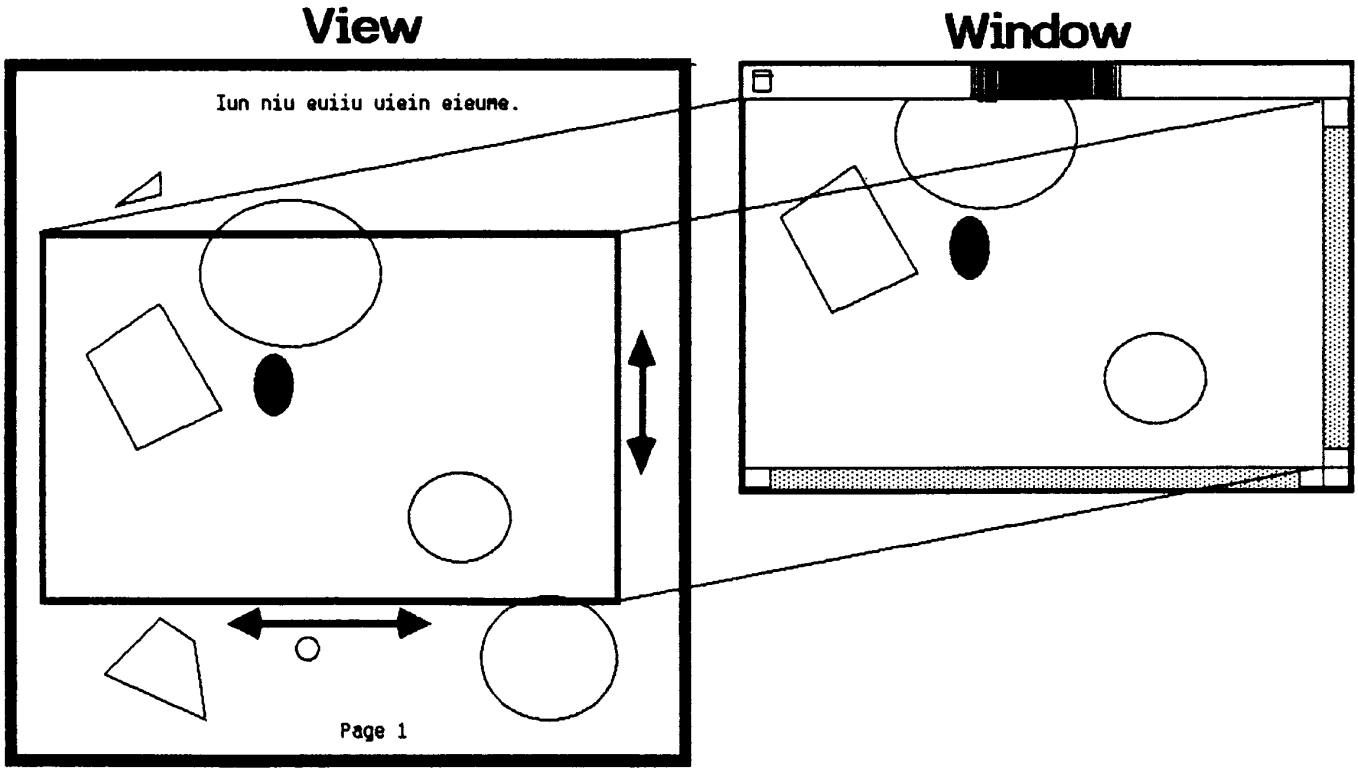
Dialog/Form (check boxes, buttons, fields)

Data base 9/15

etc.

A Window Displays a View of a Document

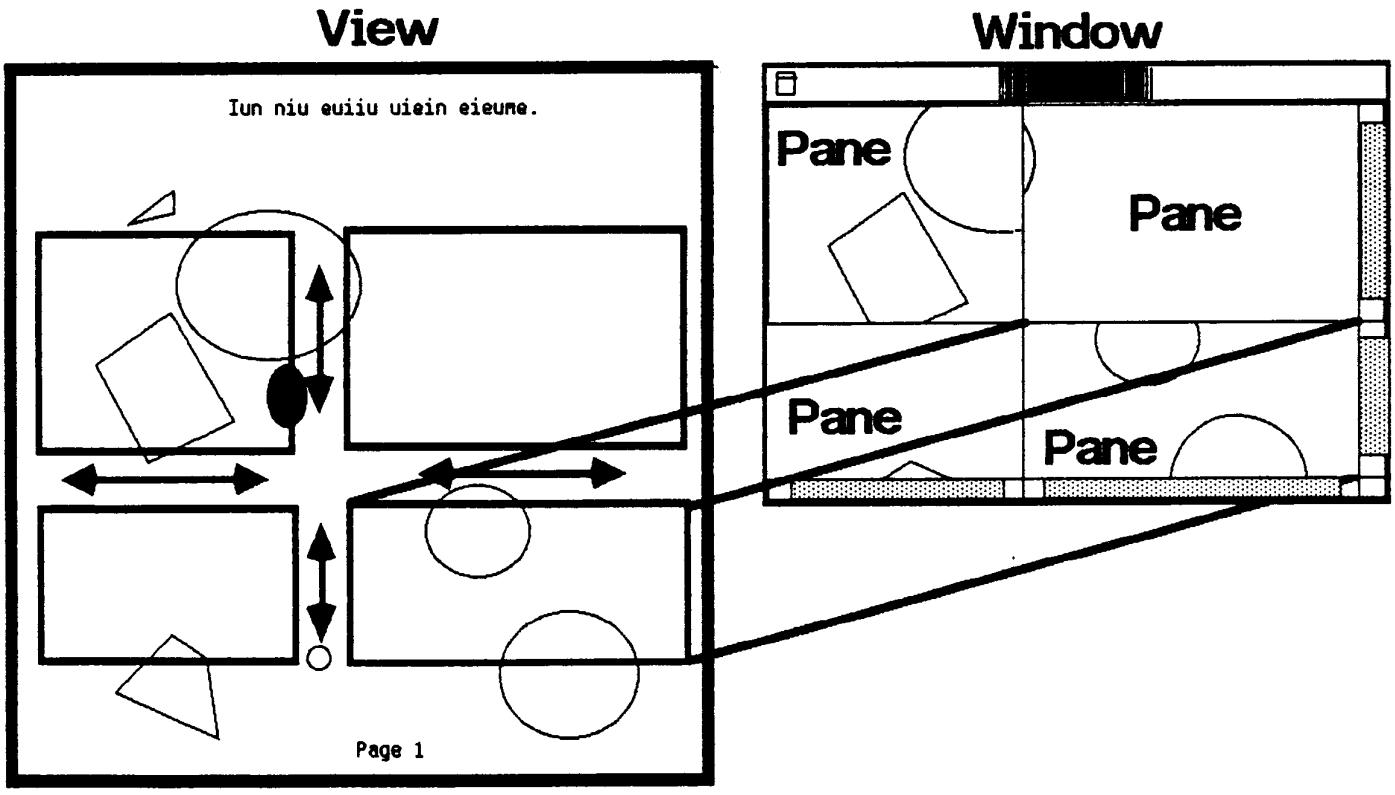
The Window can Scroll over the View



10/15

A Window can be split into Panes

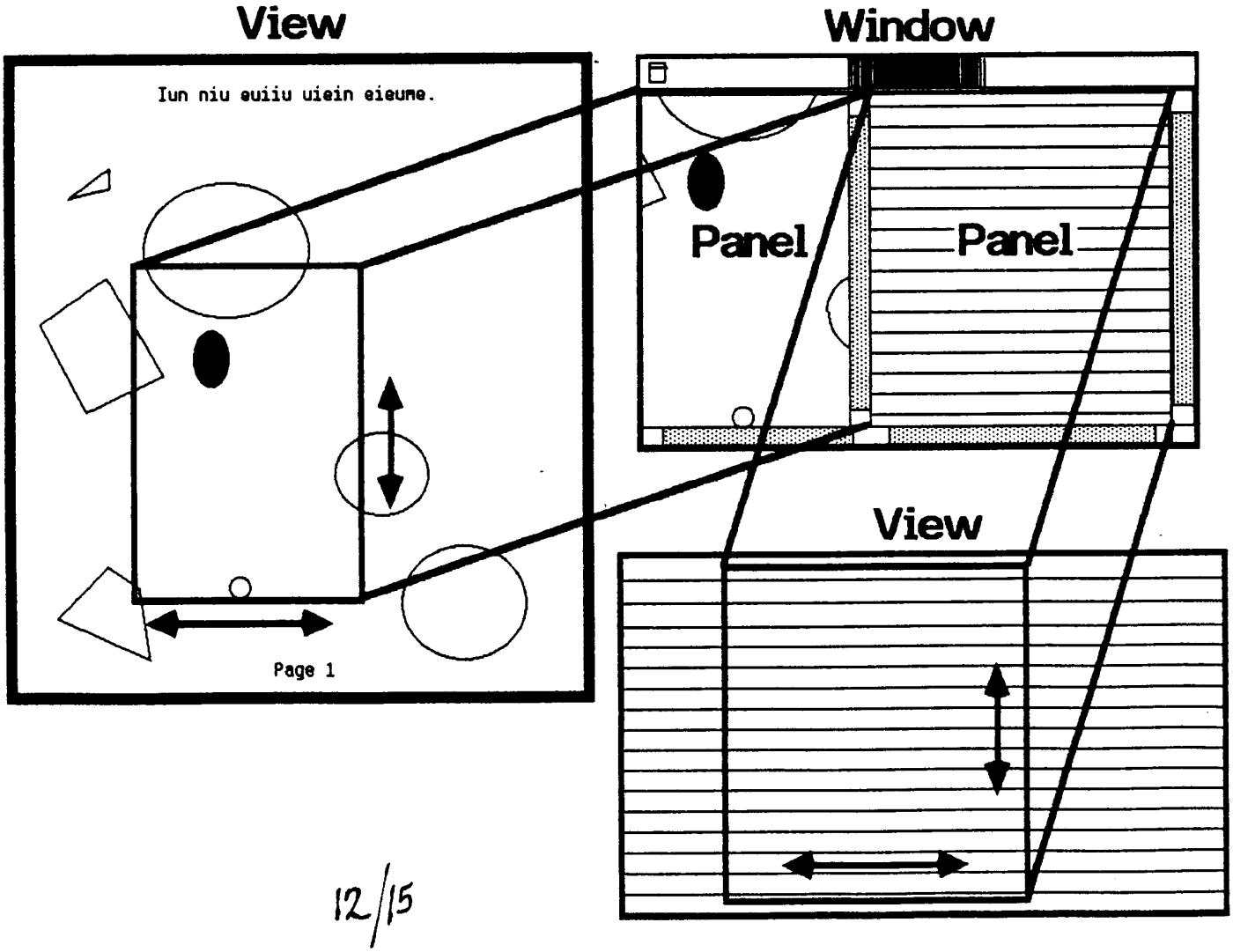
Each Pane Shows a Different Area of the View



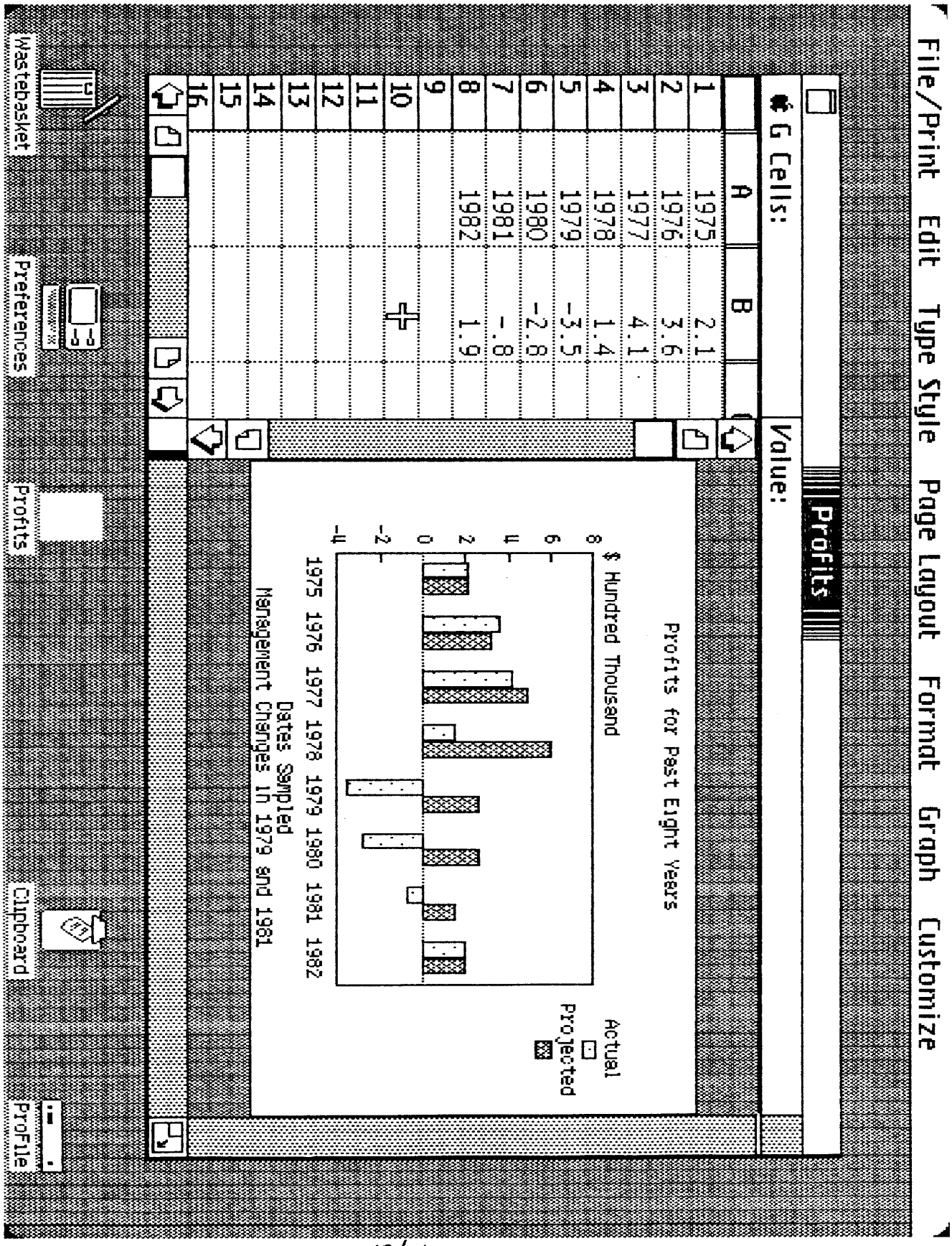
11/15

A Window Can Have Several Panels

Each Panel Shows a Different View of the Document

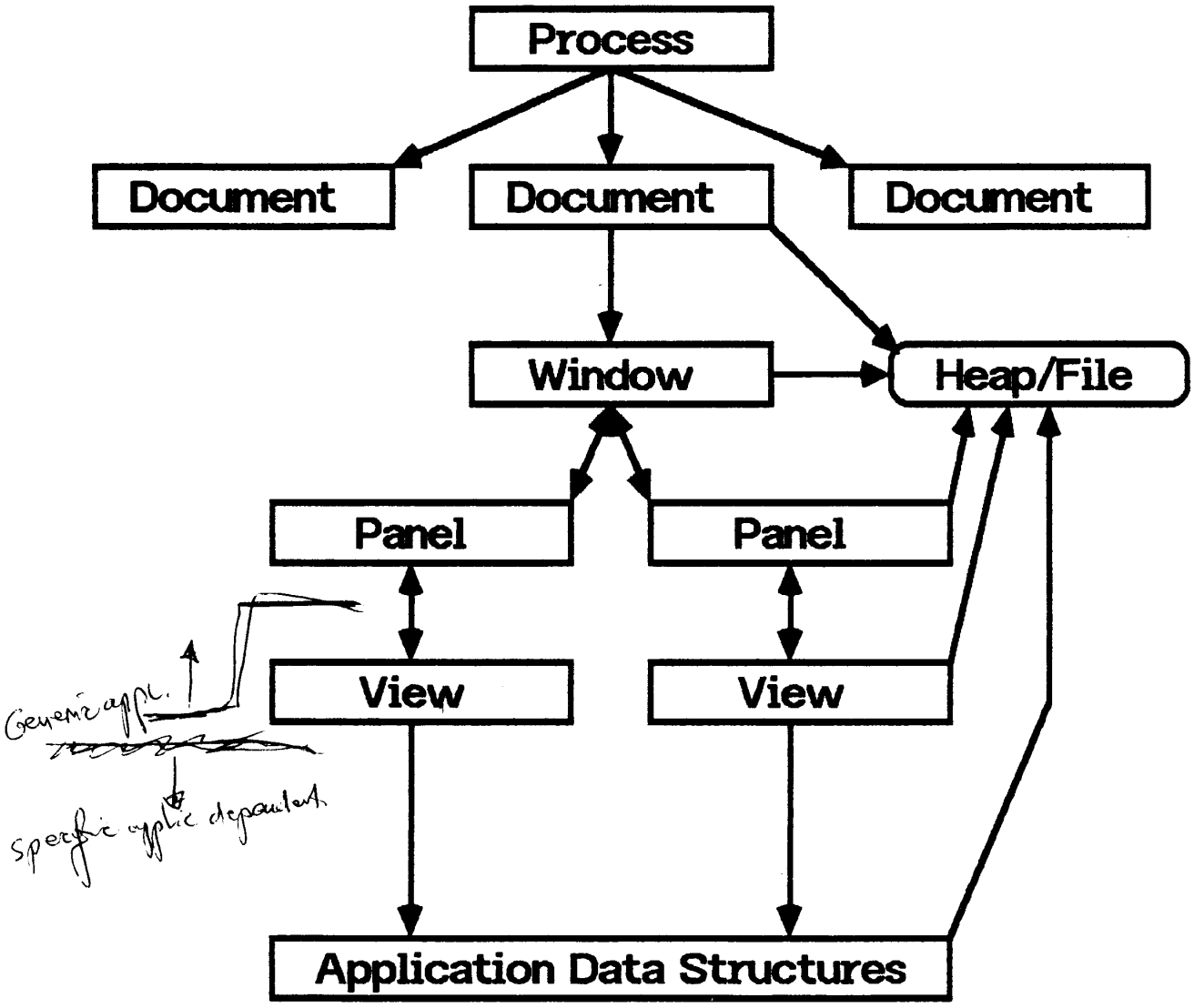


12/15


File/Print Edit Type Style Page Layout Format Graph Customize
Profits
G Cells:
Value:
A B
1 1975 2.1
2 1976 3.6
3 1977 4.1
4 1978 1.4
5 1979 -3.5
6 1980 -2.8
7 1981 -.8
8 1982 1.9
9
10
11
12
13
14
15
16
Profits for Past Eight Years
$ Hundred Thousand
8
6
4
2
0
-2
-4
Actual
Projected
1975 1976 1977 1978 1979 1980 1981 1982
Dates Sampled
Management Changes in 1979 and 1981
Wastebasket
Preferences
Profits
Clipboard
ProFile

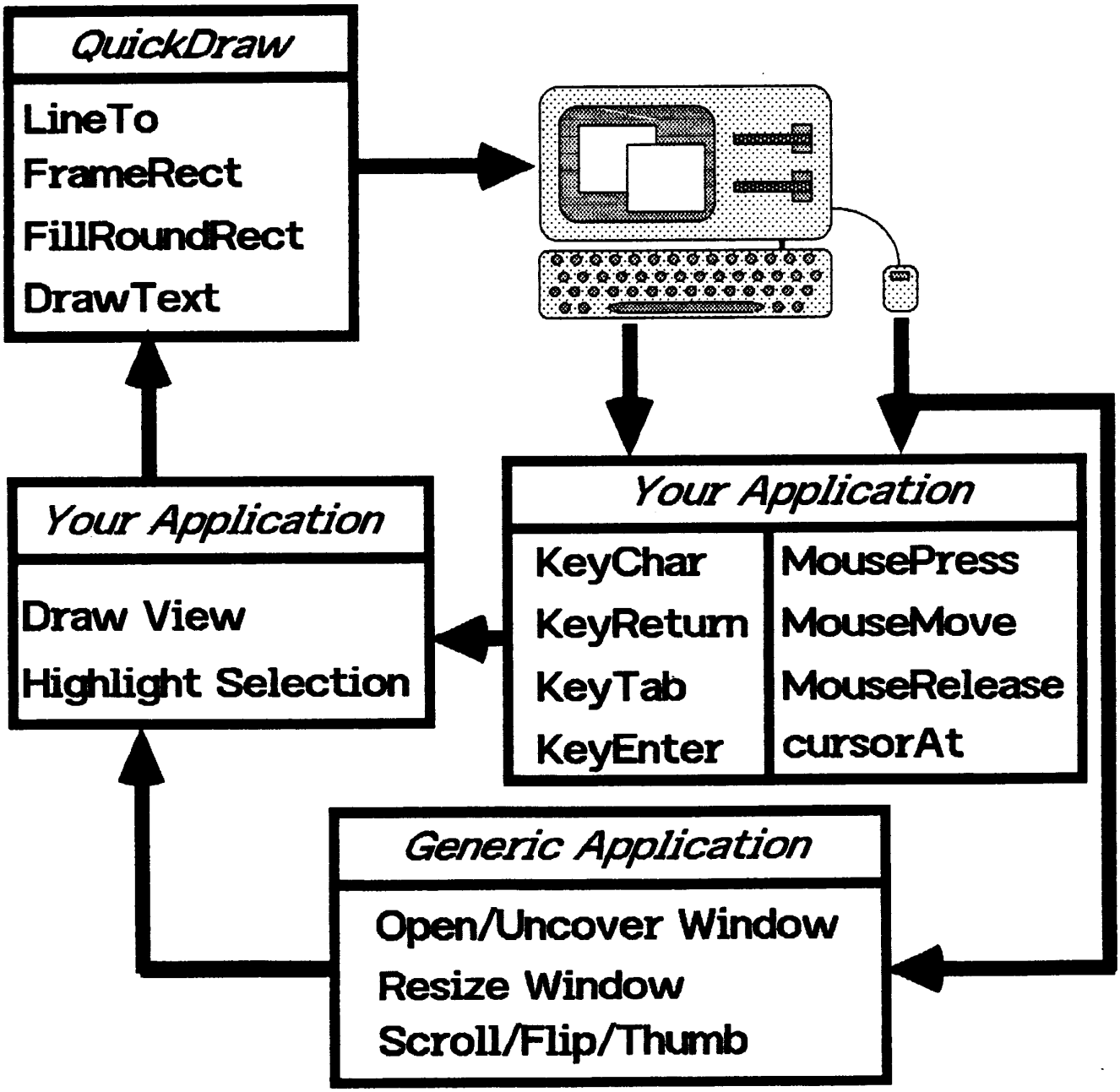
13/15

Relationships Among Tool Kit Objects



14/15

Lisa Applications are Driven by User Events



15/15