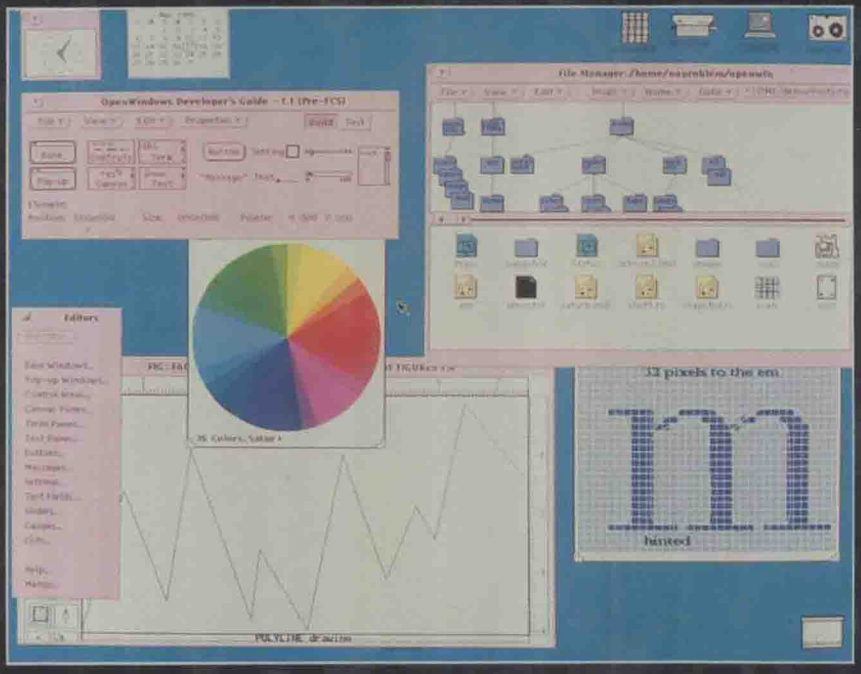
Graphical User Interfaces and Graphic Standards



Jon Peddie

COMPUTER GRAPHICS TECHNOLOGY AND MANAGEMENT SERIES

Graphical User Interfaces and Graphic Standards

Jon Peddie

江苏工业学院图书馆
藏书章

McGraw-Hill, Inc.

New York St. Louis San Francisco Auckland Bogotá Caracas
Lisbon London Madrid Mexico City Milan Montreal New Delhi
Paris San Juan São Paulo Singapore Sydney Tokyo Toronto

FIRST EDITION
FIRST PRINTING

© 1992 by **McGraw-Hill, Inc.**

Printed in the United States of America. All rights reserved. The publisher takes no responsibility for the use of any of the materials or methods described in this book, nor for the products thereof.

Library of Congress Cataloging-in-Publication Data

Peddie, Jon.
Graphical user interfaces and graphic standards / by Jon Peddie.
p. cm.
Includes index.
ISBN 0-8306-2505-4
1. Computer graphics—Standards. 2. User interfaces (Computer systems)—Standards. I. Title.
T385.P44 1991
006.6—dc20 91-18110
CIP

For information about other McGraw-Hill materials, call 1-800-2-MCGRAW in the U.S. In other countries call your nearest McGraw-Hill office.

Acquisitions Editor: Larry Hager
Book Editor: Sally Anne Glover
Production: Katherine G. Brown
Book Design: Jaclyn J. Boone

Abbreviations

APA	all points addressable
bps	bits per second
CGA	Color Graphics Adapter
cps	characters per second
DOS	Disk Operating System
DPI	dots per inch
EGA	Enhanced Graphics Adapter
EIA	Electronic Industries Association
Hz	hertz
in	inch
I/O	input/output
Kbyte	kilobytes
KHz	kilohertz
LAN	local area networks
Mbytes	megabytes
MDA	Monochrome Display Adapter
MHz	megahertz
MSB	most significant bit
ns	nanosecond
POST	Power-On Self-Test
PS/2	Personal System/2
RAM	random-access memory
ROM	read-only memory
s	second
VGA	Video Graphics Array
W	watt

Acknowledgments

The author wishes to thank the following members of Jon Peddie Associates. Without them, this book would have not been possible.

Fred Dunn

Cynthia Peddie

R. Sterling Stites

Catherine Higgins Mee

The following companies were exceptionally helpful: HP, IBM, NeXT, Nuron Data, XVT.

此为试读,需要完整PDF请访问: www.ertongbook.com

Foreword

Forewords are a bit like motherhood and apple pie. The author of the foreword very often knows the author of the book, has been involved in the field, and has responded to the author's request to provide a brief introduction to the book. I have written several forewords myself, and I have had other people write them for me. The foreword's contents are usually the result of general observations about the need for the book, and the expectation that this particular author has done a superb job of fulfilling those expectations.

This is certainly the case with Jon Peddie in *Graphical User Interfaces*. But, beyond that, this foreword is being written by someone (me) who has had a chance to not only read the manuscript, but also to make immediate use of its enormously useful contents in conjunction with some recent consulting assignments.

As a computer graphics consultant, I have been carefully following (and admittedly somewhat confused about) the details of graphical user interfaces (GUIs) and graphics standards. Certainly there is a general agreement that they are needed, but they have appeared in such profusion that it has been difficult to get a clear picture of the overall situation and to obtain details of individual offerings.

Although it is possible to scramble through much of the current periodical literature and put a moderately complete picture together, it turns out to be a lot more convenient to make use of Jon Peddie's excellent book. He has brought together an extraordinarily useful overview, detailed information about each of the offerings, and knowledgeable comments about tradeoffs in the decision-making process. I can attest first hand that the availability of this information in one volume has been extraordinarily useful to me in my work. So, while I can enthusiastically commend the book to you on a "motherhood and apple pie" basis, I can even more enthusiastically commend it to you because it is a good, solid, working document that will help you in your day-to-day trek through the intricacies of GUIs and standards. Jon Peddie is to be congratulated on an extraordinary job that needed to be done and has been done with great skill.

Carl Machover
President, Machover Associates Corporation
Past-President, National Computer Graphics Association
June 10, 1991

Preface

The purpose of this book is to identify issues affecting the choice of a graphical user interface (GUI) for the individual user and company-wide installations. The first step for most organizations that are considering adopting a GUI approach to computing is to conduct a feasibility study. This book is your feasibility study. GUIs are available for almost every type of computer and operating system on the market. This book will help you understand the choices and recognize the obstacles.

GUIs for a large organization must function in an environment of PCs, microcomputers, and mainframes. Choosing a GUI for a single platform, such as for a single user or department, is easier, but not without tradeoffs. Furthermore, you should choose a GUI that users will feel comfortable with and welcome, not one that they will rebel against or not use.

Graphics standards

After many years of frustrated experience with device-dependent software, computer users identified portability of programs from one computer system to another as the single most important objective of a graphics standard. The second objective was to have a clear-cut division between the modeling of graphics objects and the viewing of graphics images based on those objects.

A review of standards

A significant development that started in the mid 1970s was a general awareness of the need for standards in device-independent graphics packages. That emerged into a widely known specification called the Core Graphic System (Core for short). Another standard, GKS, is a 2D vector-based graphic interface with no support for bit-maps (raster operations available on modern workstations today). GKS was popular mostly in Europe. The Programmer's Hierarchical Interactive Graphics Standard (PHIGS) defines a sophisticated graphics support system that controls the definition, modification, and display of hierarchical graphics data. Additional systems such as the X Window System are being considered by standards committees.

Graphical user interface

A GUI is distinguished by its window appearance and the way an operator's actions and input options are handled. Input options to computer programs can be designed as a set of *icons*, which are graphic symbols that look like the processing option they are meant to represent. Users select processing options by pointing, with a mouse or stylus, to the appropriate icon on the screen. The advantage of these systems is that the icons can take up less screen space than the corresponding text description of the functions, can be understood more quickly if well designed, and can initiate a whole series of operations or activities. The truly unique benefit provided by a windowing system is the ability to have multiple views of different objects on the screen at the same time.

Benefits of a GUI

Recent studies have shown that users in a GUI environment work faster, more accurately and with lower frustration and fatigue levels than users in a character-based environment. Productivity has increased 35 percent, and the accuracy of completed work has improved as much as 74 percent in some environments.

Graphical user interface development

Graphical user interface systems are not a new idea. They were first envisioned by Vannevar Bush in an article he wrote in 1945. Xerox was researching graphical user interface tools at the Palo Alto Research Center throughout the 1970s. By 1983, every major workstation vendor had a proprietary windowing system. It wasn't until 1984, when Apple introduced the Macintosh, that a truly robust windowing environment reached the average consumer.

Also in 1984, out of an MIT project called Athena, arose the X Window System. Athena investigated the use of networked graphics workstations as a teaching aid for students in various disciplines. They attempted to develop a windowing system that would allow students to run local tools like word processors and spreadsheets while simultaneously being able to call up library pictures and documents from remote sources.

X Window Systems

The X Window System is a nonvendor-specific windowing system developed at the Massachusetts Institute of Technology in the 1980s. It was specifically developed to provide a common windowing system across networks connecting machines from different vendors. The X Window System (commonly referred to as *X-Windows*, or *X*) is not a GUI. It is a portable, network-transparent windowing system that acts as a foundation on which to build GUIs (such as OSF/Motif and DECwindows). The X Window System provides a standard means of communicating between dissimilar machines on a network and can be viewed in a window. Any number of windows can be open simultaneously, each potentially showing a different process on a different machine. Typically, the communication is via TCP/IP protocol over an Ethernet network. As it is with Microsoft Windows, development of the X-based GUI is ahead of application software for it.

Since the X Window System is in the public domain and not specific to any platform or operating system, it has a good chance of becoming the *de facto* windowing system of the 1990s in heterogeneous environments from PCs to mainframes. The industry trend is to adopt a fully overlapping windows system that treats the screen in an all-points-addressable (APA) or bit-mapped graphics image with soft typefonts and a mouse-driven pointer that can move around by single-pixel increments. This is clearly an idea whose time has come.

The state of things

Both operating systems and GUIs are in a tremendous state of flux. Microsoft Windows will become the dominant GUI of IBM PCs for the first half of this decade. After that, it is feasible that OS/2 will have gained enough momentum to overtake Windows in mainstream applications. Presentation Manager (PM), the GUI of OS/2, has much the same look and feel as Windows, and programs written for Windows will run under OS/2 without modification (binary compatibility). Over the long term, it seems that Microsoft Windows will continue to be more popular for 286 or 386 computers because of its relationship to DOS.

Hardware classes

Computer hardware systems can be classified into 5 general environments used to display graphic and textual information: alphanumeric terminals, personal computers, graphic terminals, graphic visualization workstations and X-Terminals. The individual characteristics are discussed in detail in this book.

How to use this book

This book is organized in as logical a fashion as possible, given the range of topics and their interrelationships. The following points should aid the reader with the organization and conventions.

Glossary The topic of standards and GUIs makes use of many new and unusual words and terms. I have made great effort to avoid arcane computer terms. In some cases it just couldn't be avoided. In all cases uncommon words and terms are explained in the glossary. The reader is advised to refer to it during reading, as some terms have come to acquire ambiguous and antonymic meanings.

Terms and conventions Even within the normal limits of grammar, protocol and laws, each book and author has a certain style. This book and author are no different. The following are the conventions used in this book.

- *Client* and *Server* are the names of specific devices or systems and will be capitalized. A person who is the client of some organization will be lower case.
- *Windows* (with a capital W) refers to Microsoft's Windows. If necessary or appropriate a version number (e.g., 2.1, 3.1, etc.) will follow.
- The word *window* (with a lower case w) refers to a window that is displayed on a computer screen.

- *Clicking* a mouse refers to pressing and quickly releasing the button on the side or top of the mouse.
- *Buttons* or *dials* on a screen refer to the symbolic representation of such items and not actually physical buttons or dials.
- Versions of programs that represent a family are designated with a small *x*, as in 3.x, or 1.x.
- Versions of a processor that refer to a family are designated with a small x, as in 808x or 80xx6, or 680x0.

Organizations mentioned All organizations mentioned in this book can be found in the Appendix. The organizations name, address and phone number are included.

Trademarks The author makes no claim to trademarked names and relies on the fair use doctrine in the use of trademarked names throughout this book.

Introduction

Graphical user interfaces and graphics standards are probably one of the most important and exciting developments of this decade. Having been in an evolutionary state since the late 1960s, thirty years later the convergence of technological and human development has made computers easier to use and more productive. This book is designed to show the choices available and guide the reader in the selection of a graphical user interface (GUI) and graphics standard.

A common user interface

The goal of many organizations is to have a company-wide computing environment that will allow any experienced computer user (regardless of skill level) to go to any available computer and use it. This is sometimes referred to as a *common user interface* (CUI). Sounds simple enough, but in practice it is seldom realized. The establishment of local area networks (LANs) made possible, among other things, access to common applications; it was a great liberator. Depending upon an individual's local setup, the computers on a LAN are often similar enough that you can access your application and/or files in another person's office or workspace. In this example the CUI is often a simple character-based menu system, or a command-line prompt that is well known.

A graphical user interface

A common graphical user interface (GUI) is another way of reaching the same goal with several added benefits. It provides a friendlier, less intimidating, and less confusing environment. A GUI environment offers familiar symbols or icons that resemble commonly known functions or items such as files and trash cans so that complicated, arcane command-line sequences do not have to be memorized. GUIs may look a bit different, but they all contain the same basic desktop metaphor of iconic symbolism. This is often referred to as the same *look and feel*.

Arguments still take place between various pedantic groups about the efficiency or speed of using a command line versus a GUI. These are much like the tavern discus-

此为试读,需要完整PDF请访问: www.ertongbook.com

sions of why a stick shift in a car is better than an automatic. For those power users who need the "feel of the machine" and don't mind wasting time trying to figure out the subdirectory structure, path, config.sys, trustee rights and/or setup on an alien machine, I say, "Have at it—may the operating system be with you." For the rest of us mere mortals who just want to get to an application, do a little work and go home or back to our own office, a GUI is a godsend.

In the 1950s, 1960s, and 1970s, scientists at the MIT Stanford Research Institute (SRI) and Xerox's Palo Alto Research Centre (PARC) investigated the interactive user interface. Their research showed that people could learn to use applications with a GUI more quickly than learning commands. So why use a GUI? Productivity, plain and simple. You can get more out of the computer in less time. You spend your time doing a job, not addressing the needs of the computer.

In the past, one of the arguments against using a GUI was the amount of power (often referred to as *cycles*) needed to make a graphic interface functional in a commercial environment. However, with the cost per MIPS (cost of cycles) steadily declining, the power argument is no longer valid.

Users are definitely attracted to GUIs. They were popularized by Macintosh, but the concept of a GUI has almost taken on a life of its own. Why is that? There is no single reason. It's due to a number of factors: a perceived ease of use (often based on hearsay), the promise of common interfaces across programs and ultimately across platforms, and an expectation that the user will have more control over the computer.

The early developers at Xerox started with a specific application area, the office. They identified problems in this domain and then applied the technology to solve them. Windows were used to expand the virtual screen and enable rapid movement among tasks. Icons were used as a reminder of what's available, and the mouse for direct manipulation to implement a simple *visual language*—where the icons are the nouns, and simple manipulations with the mouse are the verbs. Of equal importance was the need to provide the user with immediate, visual feedback about the effect of each action.

But does that mean GUIs provide the standard way of accessing computer resources, or that they're desired by every right-minded user and information systems manager? No, of course not. However, there are additional benefits that come with a GUI. According to surveys, a common quote from users and managers on GUIs versus text-based systems is that a typical GUI-based user works with six applications, while a character-based user (with a character user interface or CUI) works with no more than three or four applications. Why? Because of the more complex learning curve associated with the CUI. Users gain greater proficiency more quickly with a GUI, as shown in FIG. I-1.

For a wide range of applications, the graphical user interface style of windows and desktop metaphor is a tremendous improvement over the textual interface style that preceded it. However, although enjoying tremendous popularity now, the GUI with its icons and mouse has been around for over 15 years. The technology to support this style of interface is now maturing. Today major efforts are focused on standardizing the look and feel of these interfaces and ensuring they run in today's open system world. It's well known that the human short-term memory is limited to six (plus or minus two) items, and this constraint has a strong effect on a person's problem-solving perfor-

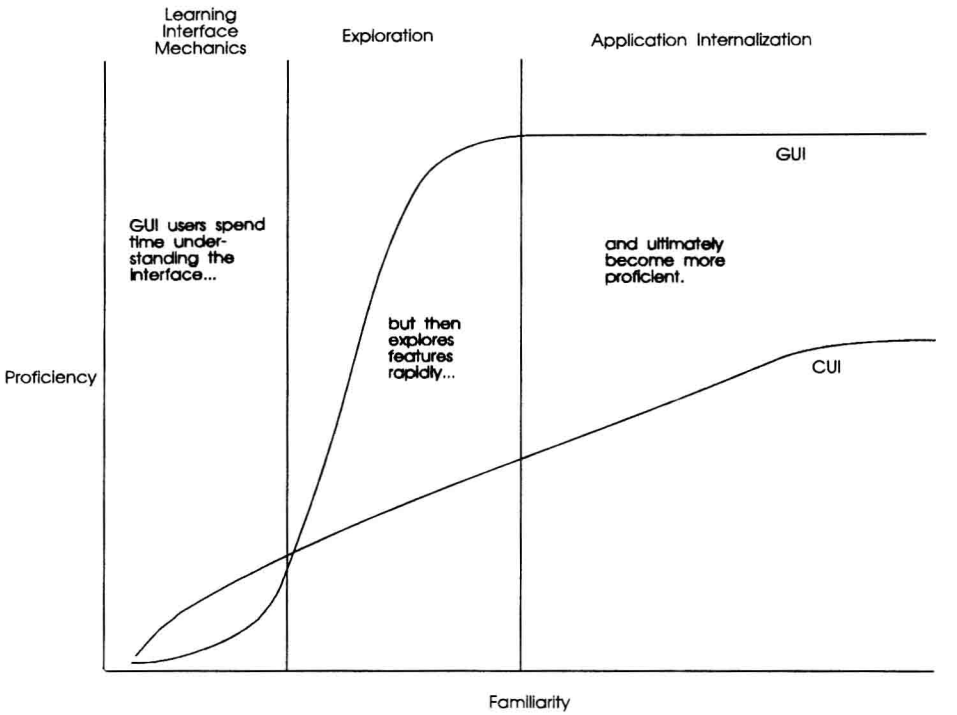


Fig. I-1. User proficiency.

mance. Visual displays act as an external short-term memory, effectively expanding the capacity beyond just six items. Visual displays also augment long-term memory as well. Icons on a screen are reminders that programs or files are available.

However, one of the real benefits isn't even graphical—it's that a GUI provides the resource to pull data from different sources and use it in a variety of applications. It would seem then that even beyond their concrete functions, GUIs seem to offer more. Some observers have pointed out that GUIs imply the future. People believe they will be more powerful and have more control with GUIs.

Enterprise vs. individual

A GUI will enhance the productivity of an individual. Equipping a user with a system that has a GUI is not very expensive relative to an organization's overall costs. As the user gets familiar with the GUI, his or her productivity will rise rapidly and then taper off. This is known as the asymptote of productivity. Put another way, for a relatively small investment an organization can see a rapid but limited gain in productivity or return on investment (ROI).

If an entire department is equipped with GUIs and true workgroup activities are employed, there will be a greater, but again limited, gain in productivity or ROI. However, the gain does not come as quickly. It takes longer to get the department working smoothly together. The training, hardware and software costs are obviously greater for

a department than for an individual. However, the ROI per employee is greater than for just the single employee. Also, the ROI curve continues to go up over a longer period of time; but, it too has an asymptote.

When an organization invests in an enterprise-wide implementation of GUIs, it takes even more time and money to see any return. However, the return is almost twice as much per employee, and it continues to increase (albeit at a slower rate) for a much longer time, as FIG. I-2 shows.

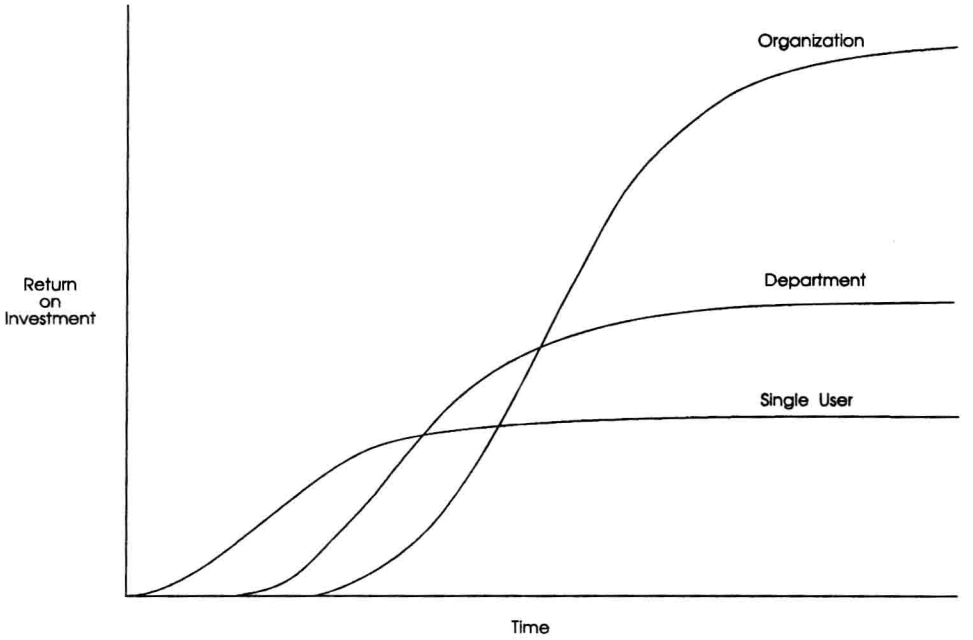


Fig. I-2. ROI for GUI.

With such tremendous ROI potential, shouldn't every organization invest immediately in an enterprise-wide GUI? Of course the answer is no. If the organization has no experience whatsoever with GUIs, all it would do is suffer an embarrassing loss and have almost nothing to show for it by plunging full force into GUIs. The logical course is to set up a test with one or two individuals. While that is in process, generate a plan, based on the empirical data gained from the test. Make no mistake, implementing an enterprise-wide GUI strategy is not a quick or easy process. The president or department manager can't simply go out and buy a dozen copies of some GUI package and expect magical results and spectacular ROI.

Enter GUI land with caution. There are riches there, but like any reward, it won't come for free.

Which one is best?

In a multivendor environment that may or may not be completely linked or networked, a GUI still offers a better chance for a new user to have some success with a machine

(assuming the chosen application is not too dissimilar). However, it is still not possible to log on to a Sun workstation and call up WordPerfect, then go to an X-terminal and do some work in Excel and finish the day in front of a Macintosh and do a dBII search. The applications simply don't run across platforms (like they should), which would provide true enterprise-wide computing. Nonetheless, the day is coming, and GUIs are the vanguard of that desirable situation. Therefore, the question of which GUI is the best one is again like a discussion about automobiles—which one is the best one?

Charm

A GUI needs computer cycles (MIPS) to operate. In the past, when computers were not as fast as they are today and when memory was more expensive, a GUI would not have been possible, or it would have operated very slowly. However, a GUI, if designed properly, brings *charm* and user friendliness to a computer. Charm uses (some say squanders) computing horsepower because it takes a lot of computer cycles and memory to make a system convenient and easy to use. Nonetheless, charm will be the deciding factor of success for computer companies and software suppliers. User charm can be translated as meaning *cheap bits* or *cheap cycles*. It takes a lot of bits and a lot of CPU cycles to meet the challenge. If those things are expensive, then they won't be available to the masses and therefore they won't deliver the promise of productivity.

A GUI that has been designed for people will have a certain feel about it, the charm factor. Although difficult to describe, you know it when you experience it. (If it were easier to describe, there would be no problem in meeting the charm specifications, and every GUI would be so endowed.) While the graphical user interface may make computing easier for users, making one work well and exhibit the charm factor creates some problems for developers—problems that did not often arise in the creation of a purely character-based application.

The developer, often not versed in the nuances of user friendliness and charm, has to make decisions on various issues concerning the design of a GUI. For example, should applications be designed with pull-down or pop-up menus? In some GUI environments, that decision is made for the developer. In others, Motif, for example, the developer has the choice of either. There are also two schools of thought on iconic representation; should they represent real objects or be a symbolic representation of their main use, suggesting the key point of the application? The symbolic icon, for example, is very popular in Japan, while in the U.S., users prefer (in general) the real objects approach—so charm has a cultural aspect as well. Those are the unresolved and somewhat difficult to define issues of charm in a GUI.

The dark side of a GUI

There are some drawbacks and precautions to take in using a GUI.

Cost "For nothing you get nothing," or put another way, "There ain't no free lunch." GUIs come with a cost, and they are far from a panacea. Because of the cost, many users will not invest in GUIs unless they can see an immediate benefit or are already using graphics-oriented programs.

Training The use of a GUI is not as intuitive or obvious as people think. The need for clear, easy-to-understand instructions will not go away just because you have a GUI.

Stress In his book, *Technostress: The Human Cost of the Computer Revolution*, Dr. Craig Brod comments on a type of stress experienced by users and programmers of computers. "We are all being socialized to be more at ease with the computer than we are with people," according to Brod. He believes those who have become used to working with computers and technology find interaction with other people stress producing.

Sociologists and psychologists are concerned about what computers are teaching us about the rest of the world. "The machine is a socialization agent that teaches you a style of interaction that is then generalized and carried over to other situations," says Dr. Tim Lynch, President of New Wave Consultants in North Quincy, Massachusetts. "Because the computer gives immediate feedback, people begin to expect immediate feedback from other people within the company," Lynch says. "They want things and they want them now." Brod says this intense man-machine relationship is both draining and addictive at the same time. "By the end of a day working with computers, you feel exhausted. There is no time for reflection. You just want to be left alone to recuperate."

Initially, a computer user is very interested in finding out about the machine and the system. As stress sets in, there is much less obsessive curiosity about what new things you can do with the computer. Some psychologists think the reason some computer users may be so stressed is that they were born with a personality prone to be stressed—a trait that may have drawn them into a career in the computer field.

Psychologists suggest setting up the work area in such a way that there is direct human interaction. Also, schedules should be arranged so that computer users have to get up, move around, and talk to their fellow employees. They also suggest more social gatherings either during or after work. They advise doing anything that brings you back to that bodily sense you have lost. If you feel the need for faster responses from the computer, some psychologists suggest that you count to 10 or get up and walk around.

The possibility of technostress is something to consider. GUIs are supposed to free us from drudgery and make our jobs easier, but often they just give us more time to do more work.

Summary

In spite (or maybe because) of the size of it, there are only three points to make in this book:

No best solution There are over a dozen GUI possibilities. Not a single one of them is the ultimate or best solution. If you think you have found the best GUI, you are lucky. But you may as well keep it to yourself, because it will only be the best one for your situation and will misdirect others.

Why use a GUI? There is one, and only one reason—productivity.

Charm If you choose to use a GUI, and it is designed right and uses the right hardware, you will know it almost instantly and benefit from its charm.

Running a computer still takes a certain amount of brain power. If you don't have or won't use that brain power, a GUI, regardless of its charm factor, won't do it for you.

References

Brod, Craig. 1982. *Technostress: The human cost of the computer revolution*. Reading, Mass.: Addison-Wesley.

Temple, Barker, & Sloane. Inc. 1990. *The benefits of the graphical user interface*. Microsoft, Zenith Data Systems: Lexington, Mass.

此为试读,需要完整PDF请访问: www.ertongbook.com