Software User Interface Compatibility and Copyright after *Lotus Development Corporation v. Paperback Software International*

Richard A. Magnan

Program on Information Resources Policy

Harvard University

Center for Information Policy Research

Cambridge, Massachusetts

A publication of the Program on Information Resources Policy.

Software User Interface Compatibility and Copyright after *Lotus Development Corporation v. Paperback Software International*

Richard A. Magnan
November 1993, P-93-4

Richard A. Magnan, Lt. Col., U.S. Air Force, is an Attorney-Legal Advisor in the Office of General Counsel, U.S. Arms Control and Disarmament Agency. This paper was prepared while the author was a U.S. Air Force Research Associate and National Defense Fellow at Harvard University.

ISBN 1-879716-04-6

Printing 5 4 3 2 1

November 1993

PROGRAM ON INFORMATION RESOURCES POLICY

Harvard University **Center for Information Policy Research**

Affiliates

American Telephone & Telegraph Co.
Applied Telecommunications Technologies, Inc.
Arthur D. Little, Inc.
Australian & Overseas Telecommunications Corp.
BellSouth Corporation
Commission of the European Communities
Computer & Communications Industry Assoc.
Corning Inc.
Dialog Information Services, Inc.
DRI/McGraw Hill
Educational Testing Service
ETRI (Korea)
European Parliament
France Telecom
Gartner Group, Inc.
GTE Corporation
R.A. Hearst
Hitachi Research Institute (Japan)
IBM Corp.
International Data Corp.
International Resource Development, Inc.
International Telecommunications Satellite Organization (INTELSAT)
Invoco AB Gunnar Bergvall (Sweden)
Japan Telecom Company
Kapor Family Foundation
Korea Telecom
Lee Enterprises, Inc.
John and Mary R. Markle Foundation
McCaw Cellular Communications, Inc.
MeesPierson (U.K.)
Mead Data Central
MITRE Corp.
National Telephone Cooperative Assoc.
NEC Corp. (Japan)
The New York Times Co.
Nippon Telegraph & Telephone Corp. (Japan)
Northern Telecom
Nova Systems Inc.
NYNEX
Pacific Telesis Group
Puerto Rico Telephone Co.
Research Institute of Telecommunications and Economics (Japan)
Revista Nacional de Telematica (Brazil)
Scaife Family Charitable Trusts
Siemens Corp.
Southam Inc. (Canada)
Southern California Edison Co.
Southern New England Telecommunications Corp.
Sprint Communications Company L.P.
State of California Public Utilities Commission
Strategy Assistance Services
The College Board
Thomson Professional Publishing
Times Mirror Co.
Tribune Company
United States Government:
- Department of Commerce
  - National Telecommunications and Information Administration
- Department of Defense
  - National Defense University
- Department of Health and Human Services
  - National Library of Medicine
- Federal Communications Commission
- National Security Agency
- U.S. General Accounting Office

U.S. Media Group
Viacom Broadcasting

Acknowledgements

The author gratefully acknowledges the very kind help of the following people who reviewed and commented critically on the draft version of this report:

Anne W. Branscomb
A. Dale Burns
Frederick E. Ellrod III
Martin L. Ernst
Robert M. Frieden
Brian Kahin
Robert H. Kohn
K.L. Laughton
Joan McGarity
Arthur R. Miller
Victor Siber
Donald J. Singer
Richard H. Stern
Frederick R. Strain
Lloyd L. Weinreb

These reviewers and the Program's affiliates, however, are not responsible for or necessarily in agreement with the views expressed here, nor should they be blamed for any errors of fact or interpretation.

I am grateful also to the members of the Program and the outside reviewers who commented on earlier drafts, and I have attempted to reflect their comments in this paper. Some outside reviewers suggested that the paper make recommendations, but the policy of the Program is that papers do not do so.

The author is especially grateful to his wife, Peggy, and he dedicates this paper to her.

Executive Summary

User interface compatibility is a complex issue involving the user and the similarity, friendliness, and consistency of interfaces. Although identical interfaces are compatible for all users, compatible interfaces need not be identical. Experienced users can readily adapt to some dissimilarities, while novice users use only a subset of the available functions and this need compatibility only for those.

Recent copyright infringement decisions have held that copyright protection extends beyond the literal source, object, and micro code to the nonliteral structure, sequence, and organization of the user interface. According to the *Lotus* court, user interface infringement occurs when an interface copies enough of the menu structure to be considered qualitatively or quantitatively substantially similar.

Neither the boundary between copyrightable subject matter and noncopyrightable subject matter (the idea-expression boundary) nor the boundary between permissible copying and copyright infringement (nonsubstantial similarity–substantial similarity boundary) is well defined. This difficulty in identifying copyrighted user interface elements and the ease and frequency of viewing them significantly increase the risk of unintentional copyright infringement for anyone who writes or modifies software for their employer.

Note

Two software applications in particular are discussed in this report, 1-2-3, a registered trademark of the Lotus Development Corporation, and VP-Planner, a registered trademark of Paperback Software International; two others are mentioned in the discussion, Quattro Pro, a registered trademark of Borland International, and VisiCalc, a registered trademark of the VisiCalc Corporation. Other brand and product names mentioned are trademarks or registered trademarks of their respective companies.

Contents

Illustration

Preface

In June 1990, the federal district court of Boston, Massachusetts, held that the user interface of Paperback Software's VP-Planner spreadsheet software infringed the copyright of the user interface of the Lotus 1-2-3 spreadsheet software.[1] As a direct consequence of this holding, Paperback Software's sales dropped from between $3 and $4 million before the suit to approximately $750,000 after the decision.[2] Paperback Software admitted copying the Lotus 1-2-3 user interface[3]: they claimed that for VP-Planner to be a commercial success, its user interface had to "conform" to that of Lotus 1-2-3,[4] but it "conformed" by copying all the Lotus 1-2-3 elements.[5] Paperback Software's claim that software development requires standardization and incremental improvement based on existing software proved an ineffective defense against Lotus's claim of copyright infringement.

If user interfaces must be identical to be compatible, and if they are copyrightable subject matter, it may be impossible to write a user interface without committing copyright infringement. If, however, nonidentical interfaces can be compatible, the issue becomes the amount of similarity required for compatibility and whether that required amount of similarity can be achieved without copyright infringement. This paper examines, first, user interface compatibility theory, to determine the extent to which two user interfaces must be similar to be compatible; then, the *Lotus* decision, to discover the extent to which a user interface may be copyrighted—that is, which of its elements can be copyrighted and which may not; and, last, it compares that extent with the extent to which compatible user interfaces must be similar.

---

[1]*Lotus Development Corp. v. Paperback Software Int.*, 740 F. Supp. 37 (D. Mass. 1990).

[2]13 *The National Law Journal*, Jul. 22, 1991, at 26 col. 4. In settling the damages aspect of this case, Paperback Software paid Lotus Development Corp. $500,000, agreed to stop marketing its VP-Planner product line by December 1, 1990, and agreed not to appeal the trial court's decision. *Lotus, Paperback Software Settle for $500,000*, Infoworld, October 22, 1991, at 5; *Paperback Pulls Spreadsheet, Won't Appeal Lotus Victory*, Computerworld, October 22, 1990, at 7. The VP-Planner product line contributed 68 percent of Paperback Software's revenue. Infoworld, October 22, 1990, at 5.

[3]Idem at 69.

[4]Idem.

[5]At least one authority suggests that interface compatibility exists only for identical interfaces. Menell, *An Analysis of the Scope of Copyright Protection for Application Programs*, 41 Stan. L. Rev. 1045, 1097, 1101 (1989).

This paper develops a theory of software user interface compatibility and applies it to one court's analysis of the applicable copyright law. The *Lotus* decision was chosen for this analysis, because it directly addresses the issue of user interface compatibility and at the time it was the most recent decision on this issue. That court's analysis has not been overridden , although subsequent cases have continued to refine copyright law as it applies to computer software.[6] An analysis of all copyright law applicable to computer software or to user interfaces is beyond the scope of this paper.[7] This paper is not a substitute for legal counsel.

Chapter Five is a "heads up" to my computer science colleagues. First, as an experienced applications and systems programmer and, then , as a lawyer, I have experienced the computer scientist's belief in the inherent right to enhance the work of my predecessors and the lawyer's knowledge of the legal bounds on that right.

---

[6]*See Computer Associates International, Inc. v. Altai, Inc.*, 1992 U.S. App. Lexis 33369 (2d Cir. 1992); *Brown Bag Software v. Symantec Corp.*, 960 F2d 1465 (9th Cir. 1992); *Lotus Development Corp. v. Borland International, Inc.*, 788 F. Supp. 78 (D. Mass. 1992).

[7]For a discussion of the copyright law in the context of the *Lotus* decision, *see Comment: Computer Software Design: User Interface—Idea or Expression?*, 60 U. Cin. L. Rev. 161 (1991).

Chapter One

Elements of a Computer Interface That Must Be Copied to Produce Compatibility

1.1 User Interface Compatibility Theory

This section discusses user interface compatibility theory, beginning with a description of the two common types of user interfaces and proceeding to an examination of how various elements or discernible entities that together constitute the user interface affect compatibility.

Surprisingly, although the literature on user interface friendliness and consistency is ample, I found none on user interface compatibility.[8] For this reason, an explanation of the interface compatibility theory used in this paper is in order. The theory is based on my experiences with a variety of interfaces used in fifteen years as an applications and systems programmer.

1.2 User Interfaces

A user interface is the means by which a person—a user—communicates with a computer. In most programs (also called applications), the user communicates with the computer through either a batch user interface or an interactive user interface, the latter either line mode or full screen mode.

1.2.1 Batch and Interactive User Interfaces

A user communicates with a computer either in a monologue (batch mode) or dialogue (line mode). The monologue is called batch mode, because the user places all instructions to the computer together in one batch and submits them for execution without further control over the computer or the intermediate results. The interactive mode is a dialogue between computer and user in which the user requests specific operations, the computer performs them, and the user then requests further operations based on the results of the previous ones.

---

[8]See *Human Factors and Interactive Computer systems* (Y. Vassiliou, ed., 1984); J. Martin, *Design of Man-Computer Dialogues* (1973); H. Simpson, *Design of User-Friendly Programs For Small Computers* (1985); 2 *Human-Computer Dialogue Design* (R. Ehrich and R. Williges, eds., 1986); *Human Factors in Computing Systems* (L. Borman and B. Curtis, eds., 1985); *Human Factors of the User-System Interface* (B. Christie, ed., 1985); *User Interfaces: Gateway or Bottleneck?* (T. Bernold, ed., 1988).

Although lawsuits have also alleged copyright infringement of a batch user interface,[9] this paper addresses only interactive user interface compatibility.[10]

1.2.2 Full Screen Mode and Line Mode

In the dialogue between user and computer, information is exchanged either in either single sentences (line mode) or in paragraphs (full screen mode). In full screen mode, the user may enter or receive data from various locations on the screen (thus, full screen) by using typed commands, cursor selections, a mouse, or some combination of them. For example, the user may select from multiple lists in different screen locations before transmitting the selections to the computer. In line mode, user input consists of typed commands entered on the line immediately following the command prompt. Unless the user can anticipate the next commands and can stack multiple commands, separated by delimiter characters on the command line, only one command at a time is transmitted to the computer.

1.2.3 Full Screen User Interface Elements

A full screen user interface consists of one or more display screens displayed in a sequence determined by user commands. A display screen consists of interface elements that display data, accept data input by the user, or provide information. Data formats within the interface elements and the interface element arrangement on the screen are the central issues in user interface copyright infringement lawsuits. Software developers spend considerable resources designing the format and placement of interface elements to maximize user friendliness and efficiency.[11] The developer's choice of interface element format and

---

[9]Synercom Technology, Inc. v. University Computing Co., 462 F. Supp. 1003 (N.D. Tex. 1978).

[10]Other user interface infringement suits include *Lotus Development Corp. v. Borland International, Inc.*, 799 F. Supp. 203 (D. Mass. 1992); *Apple Computer, Inc. v. Microsoft Corp.*, 799 F. Supp. 1006 (N.D. Cal. 1992). *Lotus Development Corp. v. Santa Cruz Operation, Inc.*, No. 90-11663-K (D. Mass.) settled out of court. *Informationweek* (Jun. 24 1991), 16.

[11]See D. Churbuck and B. Freedman, *Suits Against 1-2-3 Imitators May Have Wide User Impact, PC Week* (Jan. 20, 1987), pp. 125, 130 ("[t]he real cost of software is not in the package but in the price of training" [quoting Wayne Maples, information center consultant at the Federal Reserve Bank in Dallas]); "Interview: Apple Computer, Inc., President and Chairman John Sculley—On Fitting into the IBM World of Computing," *Personal Computing* (April 1986), pp. 145, 147 ("It's becoming apparent that the real cost is not the hardware or even the software. The real cost is teaching the user." [Quoted in Menell, *An Analysis of the Scope of Copyright Protection for Application Programs*, 41 Stan. L. Rev. 1045, 1070 n.133]).

placement from the domain of possible formats and placements may be copyrightable expression.[12]

1.2.4 Line Mode Interface Elements

The line mode user interface consists of two elements: the output data and the command prompt after which the next command or data will be entered. For this discussion of copyright infringement, the line mode user interactive interface is equivalent to a full screen mode interactive user interface with only two elements; in this paper, no distinction is made between full screen mode and line mode.

1.3 User Interface Compatibility

Computers are tools people use to increase efficiency, and, as remarked earlier, the user interface is the means by which people use computers. In order to use a computer, one must invest time and possibly money learning to manipulate the interface. The return on this investment is increased if the user can, by learning to use one interface, in effect learn to use others. These other interfaces are compatible interfaces.

One user interface is considered compatible with another when a user trained on the first can rapidly learn the second with almost equal proficiency and without new training. For example, if a user trained on and experienced with Interface A can use it to perform specific functions with a certain speed and accuracy, and if that user can perform the same functions at almost the same speed and with almost the same accuracy using Interface B, then B is compatible with A.

A user learns an interface by learning its element format and placement, command functions, and methods of selecting commands. An information element is a group of related alphabetic, numeric, or graphic information, or some combination of them, displayed on a screen. Interface compatibility measures the degree of similarity with which two interfaces present information, accept command input, and execute commands. To assure compatibility with Interface A, the developer of Interface B could copy all elements of A, but if those elements are copyrighted, the developer may infringe the Interface A copyright.

---

[12]Not all expression is copyrightable. See **section 3.1** for examples of those that are not copyrightable.

1.3.1 Interface Compatibility and Friendliness: A Comparison

Interface compatibility measures the relative efficiency with which the user can move between two interfaces. Interface friendliness (also called user friendliness), which applies to only a single interface,[13] measures how well the interface elements communicate information and accept command input.[14] A user friendly interface presents information in a manner that simplifies communication between user and computer and effectively makes the user more efficient. User friendliness may be user dependent: for the novice, a user friendly element is intuitive, while for the expert, a user friendly element is efficient.

Although interface compatibility and interface user friendliness are different, a noncompatible interface may be made compatible by increasing its user friendliness. Changes made to increase user friendliness may either destroy or restore the compatibility of interfaces. By definition, if Interface B exactly copies the information element format and placement of Interface A, then B will be compatible with A, and if Interface A is unfriendly, B will be equally unfriendly.

The developer of Interface B might avoid infringing the copyright of Interface A by copying only the noncopyrighted interface elements of A and replacing at least some of A's copyrighted information elements with newly created ones; these changes, however, in information element format and placement in Interface B may destroy compatibility with Interface A.[15]

Changes made to increase the user friendliness of Interface B might restore the compatibility with A that was lost due to changes in the information element format and placement in B which, for the user, decreased the efficiency of B below A. If the information elements of B are made more user friendly and the resultant increase in user efficiency

---

[13]See Simpson, *supra*, note 8, at 2; Vassiliou, *supra*, note 8, at 261-76.

[14]See Vassiliou, *supra*, note 8, at 29-46.

[15]It is important to emphasize that this *might* avoid infringement. Copying any amount of the copyrighted aspects of Interface A could constitute infringement.

compensates for the decrease in efficiency due to changes in format and placement, Interface B would then be compatible with A.[16]

Increasing the user friendliness of an interface can also destroy compatibility. User friendliness is increased by changing the interface element placement and format to increase the efficiency of the user, but the introduction of such changes means the interfaces are no longer identical, contravening the rule that identical interfaces are compatible. The increase in user friendliness could so alter the information element format and placement that the user would be unable to correlate the old elements with their new format and placement in the new interface.

1.4 Interface Compatibility and Consistency: A Comparison

Interface compatibility is different from interface consistency. An interface is consistent (*i*) if it contains multiple screens,[17] (*ii*) if the elements that appear in more than one screen have the same format and placement in all of these screens, and (*iii*) if the commands in each interface perform the same function in all the screens from which they can be executed.[18] Interface consistency evaluates the consistency of the placement, format, and command execution of interface elements across multiple screens in a single user interface. (Similarly, interface user friendliness, evaluates the efficiency of use of the placement, format, and command execution of interface elements on all screens of a single interface.) In contrast, interface compatibility evaluates the efficiency with which a user can apply knowledge of one interface to a second (different) interface.

Even if interface consistency were redefined so that Interface B were consistent with A because the interface elements common to both have the same format and placement and the commands common to both perform identical functions, B might still not be compatible with A. The elements of Interface A that must be copied in B to allow an experienced user trained

---

[16]One reviewer of a draft of this paper thinks there are many "trade offs" like user friendliness that can affect compatibility; another believes that Interface B becomes "more acceptable" but not compatible; and a third does not think it would be "more compatible—it simply would decrease the importance of compatibility as an operational feature or decision element."

[17]The term "screen," not window, is used here because it was used in the discussion in the *Lotus* decision. A screen may contain zero or multiple windows.

[18]See Vassiliou, *supra*, note 8, at 261-75.

on A to perform the same functions with B at almost the same speed and with almost the same accuracy have yet to be specified (see **section 1.5**). Interface consistency requires only identical format, placement, and execution for elements common to both interfaces; it does not assure compatibility, because the requirements for consistency do not address which particular elements of one interface must be copied to another to attain compatibility.

1.5 Operational and Informational Interface Elements

Interface elements do not contribute equally to interface compatibility. Operational elements are more important for compatibility than informational elements. The numerous operational elements in an interface contain the means by which the user can tell the computer to perform particular functions, such as selecting options from a list, selecting commands from a list or menu, providing space for typed commands, and providing information about running the computer, or some combination of these. All elements that are not operational are informational. These include the user's data, computational results, and information that does not pertain to the operation of the program.

1.5.1 Informational Elements and Interface Compatibility

Informational elements accept and display the user's input data, the output data created by the computer, and nonoperational information that assists the user with the data. Informational elements present numerical, alphabetic, and graphic data in a format similar to that used in nonelectronic media. Consequently, the average computer user should be comfortable reading English and working with numbers and graphic symbols in different formats and placements. Although interfaces that display informational elements in the same format and position are more likely to be compatible than those that do not, the average user should be able to adapt to some variations in language and screen placement without decreasing efficiency below what is necessary for the interfaces to be compatible. As was seen to be true of increases in user friendliness, these variations may at some point so interfere with user correlation between old and new informational elements that interface compatibility will be lost.

1.5.2 Operational Elements and Interface Compatibility

A novice user is likely to be familiar with the data to be entered into the computer but is liable to know little or nothing about the operation of a computer. One consequence is that variations in the operational elements of two interfaces can prove confusing, decreasing the

user's efficiency and, thus, more liable to destroy interface compatibility than variations in informational elements would. A user can more readily adapt to dissimilarities in information already familiar—such as those in English or in the user's data, or in the format or placement of those elements—than to dissimilarities in unfamiliar information—such as the operational elements.

Measuring linear variation in the placement of operational elements in a screen will not necessarily measure interface compatibility. For example, two word processor interfaces could both use the command bar shown below to determine whether a file being edited should be saved, deleted, copied to another file, or renamed:

SAVE DELETE COPY RENAME

After the user moves the cursor to one of the command names and presses the enter key, the computer will perform the appropriate function. In both interfaces, the cursor will appear under "S" in SAVE, and both always move the cursor to the initial letter of the next command when the user presses a cursor key once. Through training and experience, users learn that the save or delete functions can be performed without looking at the command bar—that is, they learn to save the file being edited by pressing enter or to delete the file by pressing enter after pressing the right-pointing cursor key once. In this manner, users will become equally efficient using either word processor interface, even if the location of the command bar is different in each.

In the context of this example, moving the entire command bar to any other screen location would not destroy interface compatibility. The next example shows how compatibility is destroyed when the location of the command bar is identical in both interfaces but the positions of the first two commands are switched. In the example, both word processor interfaces place the command bar at the top of the screen but Interface A uses this command bar:

SAVE DELETE COPY RENAME

and Interface B uses this command bar:

DELETE SAVE COPY RENAME

As in the previous example, in both interfaces the cursor is always positioned under the first character in the command on the left (that is, in Interface A under "S" in SAVE and in Interface B under "D" in DELETE). Both interfaces move the cursor to the first letter of the next command when the right-pointing key is pressed once. Thus, the user who selects commands based on cursor movements without looking at the screen would save the data in Interface A but would delete the data using Interface B—a result possibly disastrous when the data were to be saved. Although many programs require additional user confirmation prior to deleting data, my own experience suggests that once a function is selected, one sometimes responds affirmatively to the request for confirmation in the belief that one has indeed executed the desired function. In the example given, the user might automatically reply "yes" to a message asking for confirmation prior to deleting data, believing that the save function had been executed.

These two examples demonstrate that interface compatibility depends on and cannot be evaluated independently of the user. Because a novice user is likely to look at the command bar before selecting a function while an experienced user often selects a function without looking at the command bar, when the command bar is moved to a different screen location, as in the first example, the experienced user is not likely to be affected while the novice may have to spend time searching the screen to locate the command bar. When, as in the second example, the command bar is in the same location in both interfaces but the positions of the first two commands are exchanged, the experienced user is likely to perform a different function than the one desired while the novice may read the command bar and then select the proper one.

Changing the function of a program function key (PFK) or changing the name of the command that must be typed to run a particular function also may affect interface compatibility. Although some compatibility is lost when the user switches from a keyboard with twelve function keys located across the top to another with ten function keys at the left edge, the disruption is less than when the functions of function keys are changed.

As a systems programmer, I frequently used the IBM Interactive System Productivity Facility/Program Development Facility (ISPF/PDF) editor and utilities and the IBM Professional Office System (PROFS) electronic mail system. To scroll in a document in

ISPF/PDF, PFK 7 pages up and PFK 8 pages down, while in PROFS PFK 11 pages up and PFK 10 pages down. Thus, the page-up and page-down functions not only change keys, they also change key sequence. Usually, I remembered the appropriate pair of function keys for the interface I was using, but often from habit I tried to use the left function key to page up and the right to page down. Scrolling in the wrong direction in PROFS decreased my efficiency and increased my frustration.

Changes in the method of selecting a function can also affect interface compatibility. For example, compatibility between interfaces A and B probably would not be destroyed if both used the same command bar while A required the use of the cursor to select a particular operation while B required the mouse. If, however, A used a command bar with selection by either cursor or mouse and B required the user to type the command or its abbreviation, the user accustomed to Interface A who selects a function because of its screen position when using Interface B may not remember the necessary command name or its abbreviation.

1.5.3 Macro Compatibility

User interface macros can store and execute a sequence of commands. For example, if the following command bar is used to rename a dataset:

SAVE DELETE COPY RENAME

With the cursor under the "S" in SAVE, the user must press the right-pointing cursor key three times and then press the enter key. By combining these four keystrokes into a macro named "R" that is executed by pressing the ALT and "R" keys simultaneously, the user is saved two key strokes each time the macro is executed.

Because such macros store and execute keystroke sequences, any changes to a key's function or to the keystroke sequence required by the new user interface to perform a particular function means that macros created for the original user interface will not operate properly on the new one. Thus, the new and old user interfaces are said not to have macro compatibility. Macro compatibility is important for users dependent on numerous or extensive macros with their current interface. Even if the new user interface is user compatible, the cost of converting old macros to run on the new interface may exceed the benefits of either the new interface or its software.

Chapter Two

The Lotus 1-2-3 and Borland Quattro Pro Experiment

In my experiment learning first Lotus 1-2-3 version 2.01 and then using Borland Quattro Pro version 1.0 without any training, my purposes were (*i*) to test my compatibility theory, explained above (**section 1.1**), (*ii*) to determine whether interface compatibility requires identical interfaces, (*iii*) to simulate a user's switch from one spreadsheet application to another, and (*iv*) to learn Lotus 1-2-3 in preparation for the analysis in **Chapter Three**.

2.1 Procedure

According to the definition of interface compatibility given in **Chapter One**, Quattro Pro is compatible with Lotus 1-2-3 if a user trained on 1-2-3 can learn to use Quattro Pro rapidly and with almost equal proficiency. The experiment simulates the switch to Quattro Pro by a user experienced in 1-2-3 to determine which elements common to both interfaces are necessary for compatibility and which elements of the 1-2-3 interface are necessary for compatibility but are absent from Quattro Pro.

I chose two spreadsheet applications for this experiment, because I had never previously used any spreadsheet program (and thus had no preconceptions) and because the *Lotus* case involved spreadsheet interface copyright infringement.[19] I wanted to use the spreadsheet programs litigated in *Lotus* but was unable to obtain the defendant's software, VP-Planner. I substituted Borland's Quattro Pro for VP-Planner, because at the time Borland was currently a defendant in another Lotus copyright infringement suit.[20]

I learned 1-2-3 by reading both the Lotus manuals[21] and the book *Using 1-2-3*,[22] and by using the various 1-2-3 functions. After I felt competent with many of these functions, I manually entered the View B sample worksheet provided with 1-2-3. Then, without reading any material on Quattro Pro and without any practice with its functions, I manually entered

---

[19]*Lotus Dev. Corp. v. Paperback Software Int.*, 740 F. Supp. 37 (D. Mass. 1990).

[20]*Lotus Development Corp. v. Borland International, Inc.*, 799 F. Supp. 203 (D. Mass.).

[21]*1-2-3 Getting Started, Release, 2.01* (1986); *1-2-3 Reference Manual, Release 2.*

[22]G. LeBlond and D. Cobb, *Using 1-2-3* (1983).

the View B worksheet into Quattro Pro. Quattro Pro Version 1 can be installed with either an interface compatible with (virtually identical to that of) 1-2-3[23] or a nonidentical interface unique to Quattro Pro: for the experiment, I used the nonidentical interface. Given that both programs are spreadsheet applications, I assumed that both would perform the same functions and that the primary difficulty in switching applications would be learning to perform familiar functions using an unfamiliar interface. As I entered the View B worksheet into Quattro Pro, I recorded my impressions of the similarities and dissimilarities between its user interface and that of 1-2-3. I was able to enter the View B application into Quattro Pro without needing any documentation and with minimal assistance from the Quattro Pro help function.

2.2 Common Interface Elements I Found Necessary for Interface Compatibility

Except for an identical interface, there is no single compatible interface for all users. My evaluation is of course highly subjective, and individuals with different educations or experiences may find different interface elements necessary for compatibility. As an experienced professional computer scientist, I expect that my list of interface elements necessary for compatibility approaches a minimum set, and most users would require more similarity than I do between the two interfaces (**Chapter Four** offers a comparison of this "minimum set" with the set found copyright-protected in the *Lotus* case).

2.2.1 The Spreadsheet Cell Matrix and Its Labels

The L-shaped column and row indicators, the reverse video cursor position indicator, and the matrix of cells are essentially identical interface elements in 1-2-3 and Quattro Pro, the qualifier "essentially" allowing for inconsequential differences in format. A spreadsheet "expresses data in rows and columns and allows the user to perform operations on the data."[24] Because these interface elements are essential parts of a spreadsheet program, all spreadsheet programs should have them in some form.

---

[23]After his decision in *Lotus Development Corp. v. Paperback Software, Int.*, Judge Keeton held that the menu commands and menu structure of the 1-2-3 compatible interface in Quattro Pro infringe the 1-2-3 copyright; *Lotus Development Corp. v. Borland International, Inc.*, 799 F. Supp. 203, 223 (D. Mass. 1992). The nonidentical interface unique to Quattro Pro which I used was not at issue in that case.

[24]Lowrey, *Copyright Protection for Computer Languages: Creative Incentive or Technological Threat?*, 39 Emory L.J. 1293 n2 (citing 2 D. Chorafas, *Fourth and Fifth Generation Programming* [1986] 73-77).

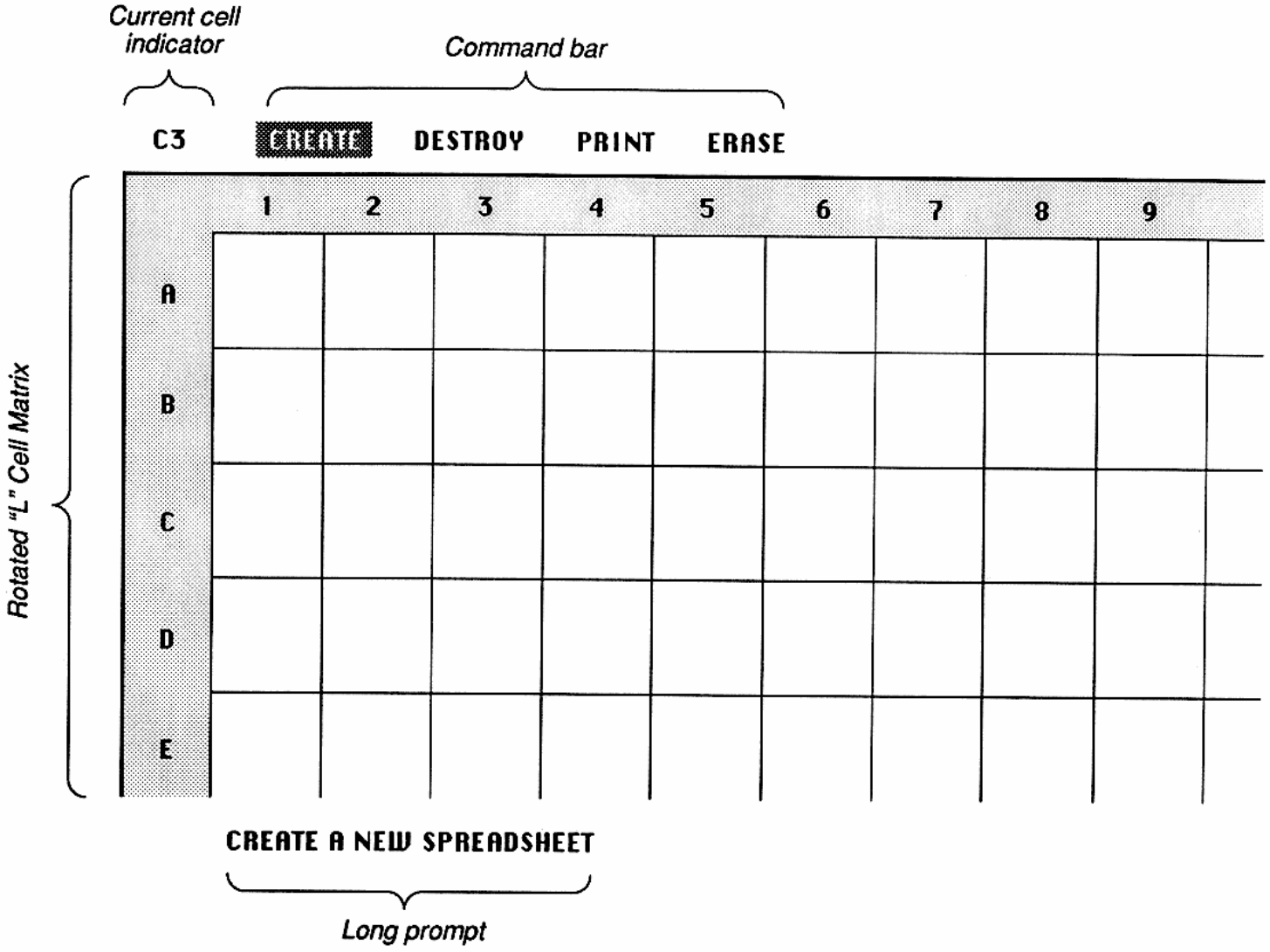

Figure 2-1

Spreadsheet Interface Elements

Their placement or format, however, can change, although I suspect such changes probably would not destroy compatibility. For example, changing the placement of the L-shaped column and row labels from the top and left side of the screen and to the bottom and right side would probably not significantly decrease the user's efficiency. After looking for the labels in the wrong screen location, the user would merely need to look to another screen location. The size of the L-shaped label and the contrast between the cursor position indicator and the remainder of the screen make locating them easy. Time lost looking in the wrong location should be so minimal that user efficiency is not significantly decreased.

2.2.2 Data Entry

Both 1-2-3 and Quattro Pro perform data input and data editing identically: the user enters data by selecting the input cell with the cursor and then typing the data. These operations are so simple that virtually any change in them would decrease user efficiency because they would increase the operation's complexity.

2.2.3 Data Formats, Labels, and Built-in Functions

The data formats, label name operations, and built-in functions are identical in both spreadsheet programs, and, as was the case with the data entry operations, the means for selecting them are probably as simple and efficient as possible. Again, any changes in them could decrease user efficiency by increasing the operation's complexity.

Differences between the two spreadsheets in the built-in function names could make it necessary for the user to consult a cross-reference chart before using a function and thereby decrease user efficiency. Short, descriptive names for the built-in function like those used in both 1-2-3 and Quattro Pro increase user efficiency, because descriptive names are easy to remember and short names require less typing. A short function name permits only a few descriptive names for a particular built-in function. If , to avoid infringement, different function names were used, user efficiency would decrease as names became less meaningful and more difficult to remember or as the time needed to type longer names increased.

2.2.4 Command Bar Access Key

In both 1-2-3 and Quattro Pro the command bars are accessed by pressing the forward slash ("/") key. I frequently accessed the command bars and suspect that my efficiency would

have decreased slightly if 1-2-3 and Quattro Pro used different keys for this operation. This change could be particularly disruptive if the command bar access key used in one spreadsheet caused a disruptive operation in the other. Learning to use different keys to activate the command bar should not be difficult unless, of course, the user were to continue to use both 1-2-3 and Quattro Pro.

2.2.5 Command Descriptions

In Quattro Pro the description for a selected command is below the cell matrix while in 1-2-3 it is located below the selected command. The difference in the placement of this interface element only affects users who cannot identify a command's function by name or by relative position in the command bar. After learning the subset of commands I frequently used, I used the command description only on the rare occasion when an unfamiliar command was needed.

2.2.6 Secondary Command Placement

In 1-2-3 the secondary commands are displayed horizontally in a command bar below the primary command bar[25]; in Quattro Pro they are listed vertically in pull-down windows that overlay the cell matrix. Because both methods display the secondary commands relatively close to their primary command, I did not notice any decrease in efficiency switching between a secondary command bar and a secondary command window. **Section 2.3.2** discusses how the format differences between the secondary command bar and secondary command pull-down windows adversely affected user efficiency.

2.3 Differences in Interface Elements That Decrease User Efficiency

2.3.1 Command Bar Display

In Quattro Pro the command bar is always displayed whereas in 1-2-3 it is only displayed after the "/" key is pressed. This difference does not affect the user switching from 1-2-3 to Quattro Pro or the user executing commands by cursor selection. However, Quattro Pro users who execute commands by command name (i.e., /FS for File Save in both Quattro Pro or

---

[25]In 1-2-3 the primary commands displayed horizontally near the top of the cell matrix are **WORKSHEET, RANGE, COPY, MOVE, FILE, PRINT, GRAPH, DATA, SYSTEM,** and **QUIT**. After the user selects a primary command, a list of secondary commands is displayed for further selection. Secondary commands can be executed only after the primary command under which they are grouped has been executed. See **section 2.3.2**.

1-2-3) and use the displayed commands as a means to remember command names will lose efficiency when switching to 1-2-3.

Although the absence of a command bar in 1-2-3 probably would not destroy interface compatibility for the user switching from Quattro Pro who is accustomed to using the displayed command bar for command names, I raise this issue to illustrate how interface compatibility depends on whether the new program provides more or less information than the old program.

2.3.2 Command Bar Format

The primary command bars in 1-2-3 and Quattro Pro have identical placement but different formats; their secondary commands differ in both placement and format. As discussed in **section 2.2.6**, these differences in placement do not decrease user efficiency below the level acceptable for compatibility. Differences in the format of the command bar cause the largest efficiency decrease and offer the most potential for destroying compatibility.

The primary command bar in Lotus 1-2-3 is:

WORKSHEET RANGE COPY MOVE FILE PRINT GRAPH DATA SYSTEM QUIT

The primary command bar in Quattro Pro is:

FILE EDIT STYLE GRAPH PRINT DATABASE TOOLS OPTIONS WINDOW

Although both command bars have the commands for **FILE**, **PRINT**, and **GRAPH**, they are in different locations within the command bar and perform some different functions. Dissimilarities exceeded similarities, and I found that my familiarity with 1-2-3's command bar was detrimental when using Quattro Pro. For example, in 1-2-3 after positioning the cursor over the RANGE primary command, I frequently used the command "FC" to Format a column for Currency. When I began working with Quattro Pro, after activating the command bar I sometimes used "FC" to format a column for currency, but in Quattro Pro "FC" is File Close, and if this program did not request confirmation before performing the operation, on many occasions I would have closed a Quattro Pro file when instead I intended to format a column.

The "FC" example illustrates the important distinction between two kinds of compatibility, function result and function operation. Although the format currency function produces identical, thus compatible, results in both spreadsheets, 1-2-3 and Quattro Pro use different (incompatible) commands to produce these results.

After using a 1-2-3 command in Quattro Pro, such as "FC," I needed to repair the undesired result before I could find and execute the desired function. The user who knows the available spreadsheet functions without knowing the 1-2-3 spreadsheet commands would first need to determine the command and then perform the function. That user would not have to reverse the effect of the incorrect command before finding and executing the correct function.

My experiment did not measure whether I was less efficient as the beginning user of 1-2-3 trying to learn both the available spreadsheet functions and their commands or less efficient as the new Quattro Pro user who knew the available spreadsheet functions but incorrectly used 1-2-3 commands to perform these functions in Quattro Pro. I suspect that the time I spent as a novice 1-2-3 user searching for the desired command was less than that spent recovering from using a 1-2-3 command in Quattro Pro.

As an alternative to learning the command names in 1-2-3, I learned the number of cursor movements required to select particular functions. For example, in 1-2-3 the input sequence "/, cursor right, enter, enter" selects the command bar containing numeric formats. When I incorrectly used this key sequence in Quattro Pro to select numeric formats, I began a cell copy operation. I then needed to exit from the cell copy operation before I could find the Quattro Pro numeric formats. The user unfamiliar with 1-2-3's command bar would begin by finding the desired function in Quattro Pro and would not need to repair undesired results.

2.4 Summary

The user interfaces of 1-2-3 and Quattro Pro are compatible but not identical. Although some differences between them decreased my efficiency with Quattro Pro, I was almost as proficient with Quattro Pro as I was with 1-2-3. This experiment shows that a compatible interface need not be identical and that changes in the placement of interface elements are less likely to decrease user efficiency than are changes in their format. The next chapter examines

the *Lotus* case to determine how different two interfaces must be to avoid copyright infringement.

Chapter Three

Elements of the Lotus 1-2-3 User Interface Protected or Unprotected by Copyright

This chapter reviews *Lotus Development Corp. v. Paperback Software International*,[26] as of July 1991 the most recent judicial opinion on software user interface copyright protection, to determine which Lotus 1-2-3 user interface elements may be copied without infringing the 1-2-3 copyright and which elements in general are copyrightable or not copyrightable.

Although the *Lotus* decision is not binding precedent for any other court and will not be refined through appellate review,[27] the court's opinion is valuable as a recent judicial analysis of the copyright statutes, the intent of Congress, and judicial precedent on whether copyright law protects the nonliteral aspects of software in general and user interfaces in particular. The *Lotus* court reviewed prior decisions on copyright protection for software's nonliteral aspects and held that nonliteral aspects are copyrightable if they meet the copyright requirements applicable to literal aspects.[28] The distinction between literal and nonliteral arises because the copyright law classifies computer software as a literary work; other literary works, such as novels, have literal words and nonliteral plots that are copyright protected. The literal aspects of software are source and object code. The nonliteral aspects of software have not yet been fully defined; they include the structure, sequence, and organization of the program and the user interface. The literal and nonliteral aspects of software are discussed further in **Chapter Five**.

Determining which nonliteral software aspects may be copyright protected is beyond the scope of this paper, which accepts the *Lotus* court's holding that user interfaces are copyrightable as nonliteral aspects and are protected by the program's copyright.[29] This

---

[26]740 F. Supp. 37 (D. Mass. 1990).

[27]The parties settled their dispute out of court. 7 *The Computer Lawyer* 30 (Nov. 1990). See also *supra*, note 2.

[28]740 F. Supp. at 54-56.

[29]The copyright office ruled that the copyright registration of the program covers the user interface and that the interface cannot be registered separately. Registration and Deposit of Computer Screen Displays, 53 Fed. Reg. 21,817-20 (1988) (to be codified at 37 C.F.R. at 202).

paper analyzes the *Lotus* decision to find which user interface elements are copyrightable and how much of that copyrighted expression may be permissibly copied.

The interface elements in 1-2-3 that the *Lotus* court found noncopyrightable are noncopyrightable because they do not meet the statutory requirements for copyrightable subject matter. The Constitution permits but does not require Congress to promote the progress of science and industry by protecting author's writings with copyrights.[30] Congress may exclude from copyright protection entire fields, such as computer programs, or specific subject matter.[31] Congress has included computer programs as copyrightable subject matter,[32] while excluding specific types of writings from copyright protection.[33]

As will be shown next, the only element of the 1-2-3 user interface that is copyrightable is so because it is not the type of subject matter specifically excluded from copyright protection. Identifying the copyrightable interface element is only the first step; it is also necessary to determine the extent of the copyrighted expression in this interface element that may be copied without infringing the program's copyright.

3.1 Elements of Lotus 1-2-3 That Cannot Be Protected by Copyright

The *Lotus* court found the following Lotus 1-2-3 elements not copyrightable: the idea of a computer program for an electronic spreadsheet, the idea of a two-line moving cursor menu, the rotated "L" screen display, the "/" key used to invoke the menu system, the enter key used to enter data into cells, and the arithmetic operators +, -, *, and /.[34] Any computer program may copy these elements without infringing the Lotus 1-2-3 copyright, because they are not the type of material protected by copyright.

---

[30]U.S. Const., art. I, §8, cl. 8.

[31]740 F. Supp. at 46.

[32]17 U.S.C. §101 (1988); H.R. Rep. No. 1476, 94th Cong., 2d Sess. 51, 54, 116, *reprinted* in 1976 U.S. Code Cong. & Admin. News, 5659, 5664, 5667, 5731 (*cited* in 740 F. Supp. at 49).

[33]17 U.S.C. §102(b) (1988).

[34] 740 F. Supp. at 65-67.

3.1.1 The Idea of a Computer Program for an Electronic Spreadsheet

The idea of a computer program for an electronic spreadsheet is not copyrightable because it is obvious and "functional."[35] Copyright law requires that copyrightable subject matter be an "original work of authorship."[36] An author's work is original if it is more than obvious; the author must add at least a modicum of creativity to the idea being expressed.[37] The court cited the VisiCalc electronic spreadsheet program, a predecessor of 1-2-3, as evidence that Lotus did not originate the idea of the electronic spreadsheet,[38] thus it cannot copyright that idea.

Even if Lotus had originated the electronic spreadsheet idea, the idea is not copyrightable because it is "functional." Copyright law denies protection to "functional" works of authorship. Ideas, procedures, processes, systems, methods of operation, concepts, principles, or discoveries are the statutory classes of noncopyrightable "functional" works.[39] "Functional" works may qualify for patent protection if they meet the more stringent patent law requirements. Patent protection requires a greater amount of novelty, nonobviousness, and usefulness than copyright protection does.[40]

3.1.2 The Idea for a Two-Line Moving Cursor Menu

The two-line moving cursor menu also is not copyrightable, because it is obvious and "functional."[41] The court found it obvious because it "is used in a wide variety of computer programs including spreadsheet programs," and "functional" because the idea for a two-line

---

[35]Idem at 65.

[36]17 U.S.C. §102 (1988).

[37]*Feist Publications v. Rural Tel. Serv.*, 1991 U.S. Lexis 1856, 10. The *Feist* and *Lotus* courts describe an expression as "obvious" if it does not contain sufficient creativity to satisfy the copyright requirement that the expression be an "original work of authorship." The copyright requirement of "originality" should not be confused with the patent law requirement of "nonobviousness."

[38]740 F. Supp. at 65.

[39]17 U.S.C §103 (1988). In this paper the *Lotus* court's term "functional" is used to mean the 17 U.S.C. §102 classes of noncopyrightable subject matter.

[40]See Maier, *Software Protection-Integrating Patent, Copyright and Trade Secret Law*, 28 Idea 13 (1987); Menell, *Computer Software*, 39 Stan. L. Rev. 1329 (1987); Milde, *Life After Diamond v. Diehr: The CCPA Speaks Out on the Patentability of Computer-Related Subject Matter*, 64 J. Pat. Off. Soc'y 434 (1982).

[41]740 F. Supp. at 65.

moving cursor menu communicates no details beyond those essential to stating the idea itself.[42] Copyright law protects an author's expression of an idea but not the idea itself.[43] If the author does no more than state the idea, there is no protectable expression.[44]

3.1.3 The Rotated "L" Screen Display

The rotated "L" screen display (see **Figure 2-1**) is not copyrightable, because it is an essential detail present in most if not all expressions of an electronic spreadsheet.[45] "[T]here is a rather low limit, as a factual matter, on the number of ways of making a computer screen resemble a spreadsheet. Accordingly, this aspect of electronic spreadsheet computer programs, if not present in every expression of such a program, is present in most expressions."[46] This legal concept, known as the merger doctrine, derives from the statutory prohibition on copyright protection for ideas.[47] If an idea has a very limited number of possible expressions, when the last expression is copyrighted, the idea itself would be copyrighted. To avoid copyrighting the idea, when an idea has very few possible expressions the law denies copyright protection to any expression.[48]

3.1.4 The "/" Key Used to Invoke the Menu Command System

The "/" key used to invoke the menu command system is not copyrightable because the idea of using an easily accessible key that has only one possible meaning merges with the idea.[49] The court recognized that, because users need to invoke this command system frequently, the key designed for this purpose must be easily accessible, must not be interpreted by the program as cell labels or cell values, and its use for this purpose should not require pressing another key (such as Shift, Alt, or Ctrl) simultaneously.[50] As a practical

---

[42]Idem.

[43]17 U.S.C. §102 (1988).

[44]Idem.

[45]740 F. Supp. at 66.

[46] Idem at 66.

[47]17 U.S.C §102(b) (1988).

[48]*Herbert Rosenthal Jewelry Corp. v. Kalpakian*, 446 F.2d 738, 741 (9th Cir. 1971) (*citing Baker v. Selden*, 101 U.S. 99, 103 [1879]). The expressions have "merged" into the unprotected idea.

[49]740 F. Supp. at 66.

[50]Idem.

matter, except for the keys used to enter labels, formulas, and values there are few keys left for the menu activation key.[51]

3.1.5 The Arithmetic Operators and the Enter Key

The arithmetic operators +, -, *, /, and the enter key used to enter data into cells are not copyrightable, because they are either "functional" "or at least obvious."[52] Without any analysis, the court acknowledged that +, -, *, and / are commonly used in computer software as arithmetic operators and that the enter key is commonly used to signal that keystroke entries should be processed.[53] Use of these keys for their commonly used functions contains no original expression.

3.1.6 Summary

The user interface elements discussed here are not copyrightable because they are the type of author's writing that Congress excluded from copyright protection. The court did not discuss whether the cell position indicator and the status indicator (located to the left and right, respectively, three lines above the rotated "L") are copyrightable. These elements, like the arithmetic operators and the enter key, probably are "functional" or at least obvious.

Having determined which elements clearly were noncopyrightable, the court then examined the "[o]ther aspects of these programs. . .[that] need not be present in every expression of an electronic spreadsheet."[54]

3.2 Elements of the Lotus 1-2-3 Copyright Protected by Copyright

The single copyrightable Lotus 1-2-3 user interface element is the menu structure taken as a whole. The menu structure includes:

- The type of menu system used (single or multiline moving cursor menu, pull down menu, or command driven interface)
- The choice of command terms (as letters, words, or symbolic tokens)

---

[51]Idem.

[52]Idem at 66-67.

[53]Idem.

[54]Idem.

- The structure and order of those terms in each menu line

- The presentation of those terms on the screen (as first letter only, abbreviations, full words, or full words with one or more letters capitalized or underlined)

- The long prompts[55] (the command descriptions listed on the second line of the two-line moving cursor menu)

Unlike the noncopyrightable elements of the 1-2-3 user interface discussed in **section 3.1**, the menu structure as a whole is copyrightable, because variations in the five subelements listed above create multiple expressions that are "not present in every expression of an electronic spreadsheet": "the command structure of 1-2-3 is an original and nonobvious way of expressing a command structure."[56]

Copying any of the five subelements of the menu structure may infringe the Lotus 1-2-3 copyright. By analogy with a copyrighted novel, however, where copying a single word is not infringement but copying the entire book is, there is a boundary between permissible copying of some copyrighted subelements of the menu structure and copying enough of them to constitute infringement.[57]

3.3 Analysis of Infringement of Menu Structure

Because the noncopyrightable user interface elements may not be copyrighted, they may be freely copied and used in any user interface. Copying any copyrighted interface elements of Lotus 1-2-3 or any subelements of its menu structure may constitute copyright infringement if the copied elements are a qualitatively substantial part of the copyrighted menu structure expression. One requirement for copyright infringement is that the alleged infringing work must be "substantially similar" to the copyrighted work; if the works are not substantially similar, there is no infringement.[58]

---

[55]Idem at 67.

[56]Idem at 68.

[57]Idem at 51-52.

[58]Another requirement for copyright infringement is that the alleged infringer must have had "access" to the copyrighted work. Copyright law prohibits copying of a copyrighted work but does not prohibit independent (i.e., without access to the copyrighted work) creation of a work identical to a copyrighted work. Independent creation is examined in **section 5.2**. This paper addresses the situation where a software developer attempts to create a user

As will be seen, the amount of Lotus 1-2-3 menu structure that may be permissibly copied will be an approximation rather than exact, because the court need only determine whether the alleged infringing menu structure, as a whole, is substantially similar to the whole copyrighted menu structure.[59] If the original work and the copy are substantially similar, infringement exists, and the court need not dissect the infringing work to determine which subelements caused the substantial similarity.[60]

3.3.1 Paperback's Admissions and Defenses

The *Lotus* court's analysis of copyright infringement determines only whether the VP-Planner menu structure is substantially similar to that of 1-2-3. Because Paperback Software admitted copying the 1-2-3 menu structure, the court did not need to determine whether Paperback Software had illegally copied the menu structure or legally independently created their own.[61] (Copyrights prohibit copying: an independently created identical work (i.e., its creator had not seen the copyrighted work) does not infringe the copyright of the original work's and may itself be copyrighted.)

Paperback Software believed that for VP-Planner "to be a commercial success [it] would have to be 'compatible' with 1-2-3."[62] "The only way to accomplish this result was to ensure that the arrangement and names of commands and menus in VP-Planner conformed to that of Lotus 1-2-3."[63] Compatibility would allow spreadsheets and macros created in 1-2-3 to be transferred to VP-Planner without either losing the macro functions or needing to retrain the user.[64]

The court cited the successful Excel spreadsheet program as proof both that exact compatibility with 1-2-3 is not necessary for commercial success and that copying the menu

---

interface that is compatible to a known copyrighted user interface by copying the known user interface.

[59]Idem at 67, 70.

[60]Idem.

[61]Idem at 69.

[62]Idem at 69 (quoting from defendant's affidavit).

[63]Idem.

[64]Idem.

structure was not the only means to achieve compatibility.[65] Paperback Software could have achieved compatibility by providing a macro conversion function and on-line help showing the 1-2-3 commands and the VP-Planner equivalents.[66]

Paperback Software's arguments for compatibility and standardization failed to influence the court's decision. Compatibility arguments "do not weigh significantly in the present decision, however, because even if VP-Planner otherwise would have been a commercial failure, and even if no other technological ways of achieving macro and menu compatibility existed, the desire to achieve 'compatibility' or 'standardization' cannot override the rights of authors to a limited monopoly in the expression embodied in their intellectual work."[67]

3.3.2 Bright-Line Rules Are Inappropriate

Paperback Software requested a "bright-line rule" for determining infringement, so software developers would "know what they may and what they must not copy."[68] They proposed replacing the idea-expression test for noncopyrightable and copyrightable subject matter with a literal-nonliteral test. Although this test could clarify the boundary between copyrightable software (source code and object code) and noncopyrightable software (all other aspects), it would not provide a bright-line method for determining substantial similarity (the boundary between permissible copying and copyright infringement). The court declined this proposal, stating that the circumstances "cry out for a judgmental standard of decision making that takes account of circumstances an impartial observer would think relevant to a 'just' disposition of the case—which, after all, is one of the declared objectives of the legal system."[69] Bright-line rules "allow for little or no evaluation and discretion, fail to take account of the competing values underlying the relevant law, and fail even to attempt to find an accommodation that serves conflicting high-value interests as well as possible, and at the

---

[65]Idem at 69.

[66]Idem at 69.

[67]Idem at 69. For a discussion of standardization with respect to the *Lotus* case, *see Comment: Lotus Development Corp. v. Paperback Software International: Broad Copyright Protection for User Interfaces Ignores the Software Industry's Trend Toward Standardization*, 52 U. Pitt. L. Rev. 689 (1992).

[68]Idem at 73.

[69]Idem at 73.

lowest possible detriment to each."[70] The court concluded by stating that Congress could have provided bright-line rules but chose otherwise.[71]

3.3.3 1-2-3 Does Not Infringe VisiCalc's Menu System as a Whole

The defendant argued that Lotus could not claim copyrights in the 1-2-3 user interface, because Lotus had copied the VisiCalc user interface rather than originating one for 1-2-3.[72] The court visually compared the menu systems of VisiCalc and 1-2-3 and found that Lotus "did not impermissibly copy copyrighted elements of VisiCalc," because "Lotus 1-2-3 uses a very different menu structure."[73]

The single command line in the VisiCalc main menu is:

Command: BCDEFGIMPRSTVW—

where each letter represents a different command: Blank, Clear, Delete, Edit, Format, Global, Insert, Move, Print, Replicate, Storage, Titles, version Number, Window, and "—" for "Label Repeating."[74] The first of the two command lines in main menu of 1-2-3 is:

WORKSHEET RANGE COPY MOVE FILE GRAPH DATA QUIT

The court noted that VisiCalc uses a single-line menu structure while 1-2-3 uses a two-line structure and that VisiCalc presents commands in alphabetical order while 1-2-3 presents commands in order of predicted frequency of use.[75] From these observations, the court concluded that the selection and arrangement of the five subelements in the menu structure of 1-2-3 is not copied from VisiCalc, but is an original and nonobvious expression of the idea for a two-line moving cursor menu.[76]

---

[70]Idem at 73.

[71]Idem at 73.

[72]Idem at 83.

[73]Idem at 67.

[74]Idem at 67.

[75]Idem at 67.

[76]Idem at 68.

The court warned that when comparing two works for substantial similarity, "a court need not—and, indeed, should not—dissect every element of the allegedly protected work. Rather, the court need only identify those elements that are copyrightable [such as the five subelements of the menu system listed above], and then determine whether those elements, considered as a whole, have been impermissibly copied."[77] Even if some of those elements are obvious or merge with the idea of a menu structure, copyrightability of the command structure taken as a whole is not precluded.[78] "If particular characteristics not distinctive individually have been brought together in a way that makes the 'whole' a distinctive expression of an idea—one of many possible ways of expressing it—then the 'whole' may be copyrightable."[79]

3.4 VP-Planner Infringes the Menu System of 1-2-3

3.4.1 Type of Menu System

Unlike VisiCalc, both VP-Planner and 1-2-3 use a two-line moving cursor menu system.[80] As already noted (**section 3.1.2**), the idea of a two-line moving cursor menu cannot be copyrighted, only the particular expression of the five subelements may.[81] The location of the menu is different in 1-2-3 and VP-Planner: in 1-2-3 the menu is above the rotated "L" display and in VP-Planner the menu is below the rotated "L" display (this program also gives the user the option of moving the menu from the bottom of the screen to the top).[82] The court merely mentioned the difference in location without indicating whether changes in the location of the menu or in its format have more effect on substantial similarity.[83]

---

[77]Idem at 67 (citing *Atari Games Corp. v. Oman*, 888 F.2d 878, 882-83 [D.C. Cir. 1989]).

[78]Idem at 67.

[79]Idem at 67.

[80]Idem at 85-87.

[81]Idem at 65.

[82]Idem at 70.

[83]Idem.

There are more possible placements for pull-down menus[84] than for those that either remain on the screen or appear on other screen space not used for interface elements. Pull-down menus, which overlay displayed information, may be placed at any screen location. The other types of menus are limited to the area outside the rotated "L" cell matrix, an area so limited that the idea of nonoverlay menus probably merges with the expression and probably cannot be copyrighted.[85]

3.4.2 Choice of Command Terms: Letters, Words, or Symbolic Tokens

Both the Lotus 1-2-3 and the VP-Planner menus use descriptive words as command terms.[86] VP-Planner also uses numbers that apparently can be substituted for the command word to activate the command's function.[87]

Perhaps infringement might not have occurred if VP-Planner had used only numbers or had used symbols rather than words to indicate the commands. The disadvantages of numbers are the decreased user friendliness—the user would need to memorize the association of the function and the number—and the loss of the descriptive aspect of command words and of macro compatibility. (Macro compatibility requires identical command abbreviations. Both 1-2-3 and VP-Planner use the first letter of the command word as the command abbreviation.) Sufficiently descriptive pictorial symbols, instead of command words, might change the overall appearance of the menu structure sufficiently to avoid infringement.[88] Macro compatibility would be retained if commands were executed by typing the first letter of the word indicated by the pictorial symbol.

---

[84]*See Apple Computer, Inc. v. Microsoft Corp.*, 799 F. Supp. 1006 (N.D. Cal. 1992) for a discussion of copyright protection for symbols (icons).

[85]In *Apple Computer, Inc. v. Microsoft Corp.*, 799 F. Supp. 1006 (N.D. Cal. 1992), the court held that, under the merger doctrine, neither pull-down menus nor split-screen menus (using nonoverlapping screen areas) may be copyrighted.

[86]740 F. Supp. at 86-87.

[87]Idem.

[88]*See Apple Computer, Inc. v. Microsoft Corp.*, 799 F. Supp. 1006 (N.D. Cal. 1992) for a discussion of the overlapping windows in Xerox's Small talk, a predecessor to pull-down menus.

3.4.3 Structure and Order of Command Terms

In both menus the command terms are listed horizontally across the screen.[89] Most lines in the VP-Planner menu begin with a help ("?") command not present in the lines of the 1-2-3 menu.[90] VP-Planner adds to the end of its command lines commands not present in 1-2-3.[91] Except for the addition of that help command ("?") and those additional commands at the end of the command line, VP-Planner includes all 1-2-3 commands, and the order of the command terms in the menus of both applications is identical.[92]

Lotus 1-2-3 lists commands by predicted frequency of use.[93] This order is probably noncopyrightable, because it is an idea or is obvious. Had Paperback Software also chosen this command order on the basis of research indicating this was the frequency of use order, they could have introduced evidence of such research at trial to prove that they independently chose the same command order. Although such proof alone might not prevent an infringement finding where the command terms are identical, in combination with other changes, especially in some or all command names, it might make the whole menu structure appear sufficiently different to prevent infringement.

3.4.4 Presentation of Command Term: First Letter, Abbreviations, Full Words, or Full Words with One or More Letters Capitalized or Underlined

In both VP-Planner and Lotus 1-2-3, the menus present the commands as full words with the first letter capitalized. In VP-Planner each command term is preceded by a reverse video number which apparently can be used instead of the first letter of the command word to select the function.[94] The court seems to view VP-Planner's reverse video number as an addition to the commands copied from 1-2-3 rather than as different expression. Copying cannot be hidden by adding elements to the copied expression: actual changes in the expression itself are required.

---

[89]740 F. Supp. 86-87.

[90]Idem at 70.

[91]Idem.

[92]Idem at 86-87.

[93]Idem at 67.

[94]Idem at 86-87.

As shown in **section 3.3.3**, the menus in 1-2-3 and in VisiCalc are not substantially similar. The court's limited analysis does not indicate whether the following menu would infringe the 1-2-3 copyright:

Command: WRCMFGDQ

This menu is patterned after the VisiCalc menu, where these are the first letters of the commands in the first line of the 1-2-3 menu. Using this menu provides macro compatibility between the two spreadsheets, because to execute a command both would use the first letter of the command name preceded by a "/". It also provides a menu structure with "aesthetic appeal" that for the "ordinary observer" would be different from that of 1-2-3.[95] If the court follows its rules and examines the two menu structures as a whole, without dissecting them to learn that the WRCMFGDQ commands are copied from the 1-2-3, there should be no infringement. If, however, the court were to decide that the choice of command activation characters (here, the first letters) is a substantial part of the menu structure, it could find infringement when these command activation characters are copied.

3.4.5 Long Prompts

Both menus use the second line to list the subcommands or a long prompt (information about a command) for a command in the first line.[96] The exact wording of the long prompt is different in 1-2-3 and VP-Planner.[97] Unless the command function described by the long prompt has so few expressions that the merger doctrine applies,[98] different wording for the long prompt changes the overall appearance of the menu but without affecting macro or user compatibility. When the second menu line contains commands instead of a long prompt, the same considerations apply as for first line commands.

---

[95]Idem at 70 (*citing* Learned Hand in *Peter Pan Fabrics, Inc. v. Martin Weiner Corp.*, 274 F.2d 487, 489 [2d Cir. 1960]).

[96]Idem at 64.

[97]Idem at 70.

[98]See *supra*, note 47 and accompanying text.

3.5 Substantial Similarity

In addition to the differences already discussed, the user interfaces in two spreadsheets differ in their start-up screens, the organization of the help screens, in the width of the screen (the VP-Planner screen is larger), and the ability to hide certain columns, as VP-Planner can but 1-2-3 cannot.[99] In spite these differences, these user interfaces are "substantially, indeed, strikingly, similar."[100] These menu structures are substantially similar from the viewpoint of the ordinary observer, who, "unless he set out to detect the disparities, would be disposed to overlook them, and regard their aesthetic appeal as the same."[101] "A laundry list of specific differences. . .will not preclude a finding of infringement where the works are substantially similar in other respects. . . ."[102] The user interfaces of 1-2-3 and VP-Planner are substantially similar, because "[f]rom the perspective of both and expert and an ordinary viewer, the similarities overwhelm differences."[103] "Moreover, even if some elements of VP-Planner were very different, it would not give defendants a license to copy other substantial elements of 1-2-3 verbatim."[104] Even if two works were 99 percent different, infringement would occur if the 1 percent copied were a qualitatively substantial part of the copyrighted work.[105]

3.6 Summary

The VisiCalc and VP-Planner infringement discussions provide examples of the requirements for copyright infringement. They define two points on the menu structure spectrum, one permissible copying (not substantially similar), the other copyright infringement (substantial similarity). The boundary between permissible copying and copyright infringement lies somewhere between the two.

---

[99]740 F. Supp. at 70. Apparently the ability to hide certain columns was added to 1-2-3 in later releases.

[100]Idem.

[101]Idem (*citing Peter Pan Fabrics, Inc. v. Martin Weiner Corp.*, 274 F.2d 487, 489 [2d Cir. 1960]).

[102]Idem (*citing Atari, Inc. v. Phillips Consumer Elecs. Corp.*, 672 F.2d 607, 618 [7th Cir. 1982], *cert. denied*, 459 U.S. 880 [1982]).

[103]Idem.

[104]Idem.

[105]Idem.

Chapter Four

After the *Lotus* Decision: Which Elements of a Copyrighted User Interface May Be Copied Without Infringing the Copyright?

4.1 Are the Lotus 1-2-3 User Interface Elements That I Found Necessary for Compatibility with Quattro Pro Protected by Copyright under the *Lotus* Decision?

Because, by the definition used here (see **section 1.3**), identical user interfaces are compatible, compatibility is assured for interface elements that can be copied without change from the copyrighted interface. Where copying an interface element would risk infringement, the element must be replaced. For compatibility, the replacement should be as similar as possible to the copyrighted element while also dissimilar enough to avoid infringement. When balancing the similarity necessary for compatibility against the dissimilarity necessary to avoid infringement, the financially prudent choice is to avoid infringement and risk losing compatibility. A copyright infringer might need to pay the owner of the copyright an amount equal to or greater than the profits obtained from the infringing work.[106]

Essential to creation of a replacement element is finding the minimum changes necessary in order to make the replacement not substantially similar to the copyrighted element. For any copyrighted user interface, multiple sets of minimum changes probably exist that might avoid infringement, each with a different effect on compatibility. The *Lotus* opinion does not explain how to find a set of minimum changes; it only provides a legal test for infringement. The new interface with the required changes can be tested for infringement by the analysis described in **Chapter Three**: identify the noncopyrightable elements, identify the copyrightable elements, and determine whether the interface containing the copyrightable elements and any replacement elements is substantially similar to the copyrighted user interface.

---

[106] 17 U.S.C. §504 (1988).

4.2 Can the Interface Elements Necessary for Compatibility Be Copied or Replaced without Destroying Compatibility?

4.2.1 The Cell Matrix and Its Labels

The rotated "L" screen display (see **Figure 2-1**) may be copied without infringement because it is not a copyrightable element of an electronic spreadsheet program (see **section 3.1.3**).

4.2.2 Data Entry (Input and Editing)

4.2.2.1 Data input. Data input requires positioning the cursor on the desired cell, typing the data, and pressing the enter key. The enter key as a means to enter data cannot be copyrighted (see **section 3.1.5**). Although the *Lotus* court did not discuss cursor positioning or typing data, these parts of data entry, like the enter key itself, are probably obvious or noncopyrightable subject matter that can be copied without infringement.

4.2.2.2 Data editing. The two editing methods, positioning the cursor and reentering the data, or positioning the cursor and using the F2 key to edit the cell contents, were not discussed in the *Lotus* decision. Positioning the cursor and reentering the data merely comprise data entry performed on a nonempty cell (that is, one that contains data the user previously placed there; an empty cell, in contrast, either had no data placed there or the data were deleted); this method is noncopyrightable for the same reasons that data entry is noncopyrightable.

Positioning the cursor and using the F2 key to edit the cell data may, like data entry, be obvious or noncopyrightable subject matter or may have copyrightable expression in the use of the F2 key rather than one of the other function keys. Whether the *Lotus* court would find copying this function key permissible given that, as with the menu activation key ("/"), there are few practical alternatives (see **section 3.1.4**), is not clear. Or, the court could find the choice of edit function key copyrightable because, unlike potential menu selection keys, function keys are equally accessible, equally unambiguous, and sufficiently numerous to provide ample alternate expressions for the idea of using a function key to enter edit mode (see **section 3.1.4**). If using a different edit function key were necessary to avoid infringement, user efficiency might decrease slightly until the user learned the new edit key.

4.2.3 Data Formats, Labels, and Built-in Functions

4.2.3.1 Data formats. Data formats are the various numeric formats (i.e., currency, fixed, scientific) and label prefix characters (apostrophe for left justification, quotation marks for right justification, caret for centering) that determine how the data in the cells are displayed on the screen and on a printout. Although the *Lotus* court did not specifically discuss these elements, its holding that the rotated "L" cell matrix is noncopyrightable could include data formats, because there are few ways of displaying numbers and data in each of the various formats.

Alternatively, the court's holding could apply only to an empty cell matrix. A cell containing formatted data provided by the user might have copyrightable expression in the choice of data and cell format. (It could be argued that the user who provided the data and selected the cell format owns the copyright in the formatted cell; or, it could be argued that the copyrightable expression in the cell format was determined when the software was written, because the software displayed the formatted cell after combining the data and the cell format descriptor.)

The numeric formats alone, without any data provided by the user, are probably obvious or "functional" like the arithmetic operators and therefore are noncopyrightable (see **section 3.1.5**). The label prefix characters alone, without data, are, like the menu activation key ("/"), probably noncopyrightable because few other symbols with unambiguous meanings are available (see **section 3.1.4**).

4.2.3.2 Labels. Label name operations (that is, the assignment of a name to a block of cells in a spreadsheet and the subsequent use of that name during operation instead of specification of the cell addresses in the block), like the rotated "L" display, are probably noncopyrightable aspects of the electronic spreadsheet idea present in most expressions or noncopyrightable "functional" subject matter (see **section 3.1.2**). All electronic spreadsheets should be able to provide these label operations without infringement.

4.3.2.3 Built-in function names. The *Lotus* court did not discuss built-in functions. They are not part of the menu system and not displayed on any user interface screen, so whether they would be copyrightable as either literal or nonliteral aspects of the

program is not clear. The court could decide that as nonliteral aspects, few names are possible for the built-in functions and could find by analogy to the noncopyrightable "/" menu activation key that common functions (e.g., mathematical functions with common mathematical names) are obvious, like the arithmetic operators; or by analogy between built-in function names and the command terms of the menu system, it could find them copyrightable. Alternatively, the court could find the names of the built-in functions protected as literal source or object code, because they appear in the code as names of subroutines or entry points. (A court could use more than one of these methods: it could find common mathematical functions with common mathematical names obvious (thus not copyrightable), like the arithmetic operators; or, it could find that some functions have so few possible descriptions that, like the "/" menu activation key, they are noncopyrightable; or, it could find other functions copyrightable as having nonobvious names with ample alternatives.)[107]

If the names of the built-in functions needed to be changed to avoid infringement, the result would affects compatibility adversely in two ways. First, the user would need to learn the new names; a help function consisting of a cross reference list of the new and old names for the functions would ease this difficulty.[108] Second, spreadsheets containing the names of the built-in functions would not be portable to another spreadsheet program, unless a utility existed to convert the old names for these functions to the new ones. Such a program, analogous to the macro conversion program suggested by the *Lotus* court, could be provided.[109]

4.2.4 Command Bar Access Key

The "/" key used to invoke the menu command system may be copied without infringement, because it is not a copyrightable element of an electronic spreadsheet program (see **section 3.1.4**).

---

[107]Although there were no trademarks at issue in the *Lotus* case, there could be trademark implications involved with the copying of unique function names.

[108]The *Lotus* court suggested using an on-line help function to list the 1-2-3 commands and their VP-Planner equivalents. 740 F. Supp. at 69. A similar feature could list the built-in functions of 1-2-3 and their equivalents in VP-Planner.

[109]Idem at 69.

4.2.5 Command Descriptions ("Long Prompts")

The command descriptions, or "long prompts," are a copyrightable subelement of the menu system as a whole (see **section 3.2**). Given that all electronic spreadsheet programs perform similar functions and that there are few brief descriptions of these functions, the courts will probably not require much dissimilarity in the descriptions to avoid infringement. Long prompts are most helpful to users unfamiliar with the commands; experienced users probably rarely use them. If the long prompts adequately describe the commands, slight dissimilarities in wording should not significantly decrease user efficiency.

4.2.6 Location of Secondary Commands

The location of secondary commands is a copyrightable subelement of the menu system (see **section 3.2**). Moving the commands (without changing the their names) from a two-line moving-cursor menu to pull-down menus did not decrease my efficiency. To select a command, I only needed to switch from the left and right-pointing cursor keys to the up and down-pointing keys (see **section 2.2.6**). Avoiding infringement by replacing the copyrighted type of menu with a different type without changing the names of the commands should not destroy interface compatibility.

4.2.7 Command Bar Display

Because the menu structures in both 1-2-3 and VP-Planner are displayed only after the "/" key is pressed, the *Lotus* court did not need to address whether a permanently displayed menu structure, like that in Quattro Pro, is copyrightable. Even if this aspect of a menu structure were copyrightable, it is only one subelement of the structure as a whole and probably not so influential in the infringement analysis as the five other subelements listed by the court. As discussed in **section 2.3.1**, the absence of a command bar affects interface compatibility in so few situations that it should not significantly affect compatibility.

4.2.8 Command Bar Format

Of all the requirements for interface compatibility discussed in **Chapter Two**, differences in the format of the command bar cause the largest decrease in efficiency and offer the most potential for destroying compatibility (see **section 2.3.2**) by affecting both user compatibility and macro compatibility. Three of the five copyrightable subelements of the menu structure (choice of command terms, structure and order of command terms, and presentation of

command terms on the screen) apply to the format of the command bar. Compatibility depends almost entirely on the extent of the difference that would avoid infringement between the new format of the command bar and the copyrighted format. The *Lotus* court did not discuss this issue.

The *Lotus* decision tells us only that the menu structures of VisiCalc and 1-2-3 are sufficiently different to avoid infringement while the menu structures of VP-Planner and 1-2-3 menu structures are not. The unfriendly command bar in VisiCalc requires that the user remember the available functions and their command abbreviations. If VP-Planner used a VisiCalc style command bar that contained the first letters of the 1-2-3 commands in the same order as in the menu structure of 1-2-3, and if this structure did not infringe the 1-2-3 copyright, the user interface would probably be incompatible for all except experienced users who could select functions by cursor position or knew the 1-2-3 primary and secondary commands well enough to select them by their first letter (see **section 3.3.3**).

4.3 The Quattro Pro User Interface

I found the Quattro Pro user interface compatible with that of 1-2-3 and believe it would remain compatible even if the long prompts, the edit function key, and the names of the built-in functions were changed to avoid infringement.[110] If the Quattro Pro interface does not infringe on that of 1-2-3, it can exemplify a noninfringing compatible user interface somewhere on the spectrum between that of VisiCalc and that of VP-Planner.

The *Lotus* opinion does not explain which changes to the subelements of the menu structure would avoid infringement, but, instead, leaves this subjective decision to the judgement of the impartial observer, taking into account circumstances relevant to a just disposition of the case.[111] The court warned that the menu structure must be evaluated as a whole without dissection into its five subelements and comparison of them individually.[112] The following comparison of 1-2-3 and Quattro Pro should illustrate some of the types of

---

[110]I do not mean to imply that the Quattro Pro user interface infringes that of 1-2-3; I am simply using it to illustrate changes that would not destroy compatibility. I used the Quattro Pro unique interface, not the 1-2-3 compatible interface. See *supra*, note 23 and accompanying text.

[111]740 F. Supp. at 73. See **section 3.4.5**.

[112]Idem at 67.

changes in the subelements of the menu structure that may avoid infringement for any user interface.

The first 1-2-3 command menu appears as:

Worksheet Range Copy Move File Print Graph Data System Quit

Global,Insert,Delete,Column,Erase,Titles,Window,Status,Page

The first Quattro Pro command menu appears as:

File Edit Style Graph Print Database Tools Options Window

Lotus 1-2-3 uses a two-line moving-cursor menu, Quattro Pro a single-line one, and both use words as commands with the first letter capitalized but some command names differ and identical commands are presented in different order. In 1-2-3 the secondary commands are displayed below the primary command line, while in Quattro Pro secondary commands are not displayed until the primary command is selected and then they appear vertically in pull-down menus that overlay the rotated "L" cell matrix.

Long prompts are displayed below the primary commands in 1-2-3 and below the cell matrix in Quattro Pro. Some long prompts for identical primary commands differ. The Print primary command long prompts are "Send print output directly to printer" in 1-2-3 and "Print a spreadsheet or graph" in Quattro Pro. The long prompt for the Graph commands are "Create a graph" in 1-2-3 and "Display/Print business graphics from spreadsheet data" in Quattro Pro.

Lotus 1-2-3 and Quattro Pro use different names for secondary commands for the identical primary commands. In 1-2-3 the secondary commands of the File command are Retrieve, Save, Combine, Xtract, Erase, List, Import, and Directory; in Quattro Pro they are New, Open, Retrieve, Save, Save As, Close, Close All, Erase, Directory, Workspace, Utilities, and Exit.

The differences between the 1-2-3 and VP-Planner user interfaces were insufficient to prevent infringement (see **section 3.4**). Whether the differences between the 1-2-3 and Quattro

Pro user interfaces noted above are sufficient to avoid infringement depends on whether, from the perspective of expert and ordinary users, the similarities overwhelm the differences.[113]

4.4 Interface Compatibility Theory Revisited

4.4.1 Why Develop Compatible User Interfaces?

Software is a tool, as noted at the outset, that increases productivity and is acquired if the expected profit from increased productivity exceeds the costs of acquisition, training, and conversion (of macros or other software resources necessary to run on the new application).

4.4.1.1 Training costs. Training costs could be eliminated if the user interface of the new software were identical to one users already know, and they would be reduced if the interfaces of the new and the known software are compatible. Such reductions would benefit both purchaser and seller (or software developer). The purchaser receives efficiency increases otherwise too expensive to implement, and the software developer profits from selling software that otherwise would be unsold.

The *Lotus* court held that user interfaces are protected by the program's copyright and that writing a user interface identical to one that is copyrighted is copyright infringement.[114] Writing compatible, noninfringing interfaces becomes the primary method of reducing training costs.

4.4.2 What Is Compatibility?

As already noted, two interfaces are compatible if a user familiar with one can rapidly learn to use the other with almost equal proficiency, and identical interfaces are, by definition, compatible for all users—those familiar with one can use the other with equal efficiency. Again, as noted earlier, for nonidentical interfaces, compatibility is complex and depends on the user, the known interface, and the new interface. Compatibility could be seen as a binary issue—it exists or it doesn't—or as a matter of degree—the degree of "rapidity" or of "almost equal proficiency," or both, with which the user operates the new (compatible) interface.

---

[113]Idem at 70. See **section 3.5.4**.

[114]Idem at 76-77, 80-82.

If compatibility is binary (binary theory), user interface compatibility exists if the anticipated profit from the new software is greater than the acquisition cost.[115] If a matter of degree (degree theory), compatibility increases or decreases as the ratio of anticipated profit to acquisition cost increases or decreases. Regardless of the theory used, the most uncertain component in the profit/acquisition cost ratio remains the training cost for a nonidentical user interface. This cost will depend on both the user and the amount of dissimilarity between the known and the new user interfaces.

4.4.3 User Knowledge as a Subset of Available Functions: Five Classes

For a particular software program and user, the functions of the program can be grouped into five classes[116]:

(*i*) Functions understood; ready to use

(*ii*) Functions understood; ready to use; but not worth using

(*iii*) Functions the user "plans to learn," but not yet understood

(*iv*) Functions considered not worth learning, because not likely to be valuable

(*v*) Functions irrelevant; no effort to learn; not valuable

For a particular user, a compatible interface need only be compatible for the functions in the first class. The other functions are not known (thus no training costs) or not used (no need to learn the equivalent functions in the new interface).

4.4.4 Degrees of Compatibility for a Particular User

A compatible user interface reduces training costs by making the user's knowledge and experience from the old interface applicable to the new one. For a particular user, compatibility can be evaluated for each class (*i*) function. The degree of compatibility equals the amount of knowledge from using the old interface directly applicable to the new one and can be grouped, by function, into four classes:

---

[115]The comparison could have been "greater than or equal to," but a business would not acquire something that does not produce a profit.

[116]This list of classes is an extract from Figure 1 ("Use of Major Applications Functions") in Martin L. Ernst, *Users and Personal Computers: Languages and Literacy, Costs and Benefits* (Cambridge, Mass.: Harvard University Program on Information Resources Policy, 1992, P-93-1), 28.

(*i*) If both interfaces perform the function identically, no training is required and no efficiency based on experience is lost. The user operates the new interface exactly the same way as the old one.

(*ii*) Functions are similar in both interfaces but not identical. The user's knowledge of the functions must be supplemented by learning something, but not everything, about the functions of the new interface.

(*iii*) The function in the new interface is so different from that of the old that knowledge of the old interface neither helps nor hinders use of the new one. The user must learn everything about performing the function in the new interface before using it.

(*iv*) Knowledge of the function in the old interface is detrimental to using the function in the new one. Old knowledge induces the user to try incorrect procedures before realizing that this knowledge is not helpful and that everything must be learned about performing the function in the new interface.

4.4.5 Degrees of Compatibility by User

The likelihood of substantial similarity between a nonidentical compatible interface and a copyrighted interface decreases as the number of interface elements that are similar in both interfaces decreases and as the similarity between similar interface elements decreases. As already noted, to avoid infringement, the elements of the new user interface must not be substantially similar to copyrighted elements in the copyrighted interface, viewed as a whole.[117] On the other hand, for compatibility, the new interface should have as many elements similar to those in the copyrighted interface as possible and they should be as similar as possible.[118] Because substantial similarity measures qualitative and quantitative similarity, in avoiding copyright infringement by avoiding substantial similarity, the designer of the compatible interface must balance the number of elements similar to the copyrighted elements, against the amount of similarity between elements of the new and the copyrighted interfaces. Again, as noted, a nonidentical user interface may not be compatible for all users (binary theory), or it may not be equally compatible for all users (degree theory).

---

[117]740 F. Supp. at 67-68.

[118]Or, as discussed in **section 1.1**, user friendliness or consistency enhancements could substitute for similarity of elements. For clarity, this discussion considers only similarity as a means to attain the user efficiency necessary for compatibility. Increases in user friendliness or consistency are means to attain compatibility with fewer similar interface elements or less similarity of interface elements, or both.

A novice user's small set of known and used functions from the old copyrighted interface requires compatibility only for a small part of the new one (see **sections 4.4.2** and **4.4.3**). These elements should be very similar, or the novice's limited knowledge will prevent recognition of them as equivalent elements. The remaining elements can be totally dissimilar, because the novice user has yet to learn them for either interface.

For an expert user compatibility requires relatively many similar interface elements (because the set of known and used functions is large) but these do not need to be very similar to the copyrighted elements. The expert's greater experience and knowledge of the old user interface permits ready recognition of equivalent features in the new interface. For the expert, the compatible new interface will contain relatively many elements similar to those in the copyrighted old interface, but they will not be so similar or will be less similar those in the novice's compatible interface.

Chapter Five

How Does Copyright Protection for Nonliteral Aspects of Software Expose Software Developers to Increased Risk of Copyright Infringement Liability?

5.1 The Difference Between Literal and Nonliteral Software Copyright Protection

For the purpose of copyright, computer software is bisected into literal and nonliteral aspects. The literal aspects, the source code, object code, and microcode,[119] comprise the "set of statements or instructions to be used directly or indirectly in a computer in order to bring about a certain result."[120] It has been settled that these aspects are protected by copyright.[121]

The nonliteral aspects of software include the structure, sequence, and organization of the program,[122] the text, layout, graphics, and other aspects of the graphical user interface,[123] and the menu structure as a whole.[124] This list may change as the extent of copyright protection for software's nonliteral aspects is debated and litigated.[125]

This legal distinction between literal and nonliteral software aspects arises from the decision by Congress that computer software may be copyrighted as a literary work,[126] which are "works, other than audiovisual works, expressed in words, numbers, or other verbal or numerical symbols or indicia, regardless of the nature of the material objects, such

---

[119]Goldberg and Furleigh, *Copyright Protection for Computer Programs: Is the Sky Falling?* 17 Am. Intellectual Prop. Assn. Q. 158 (1989). See also A. Clapes, *Software, Copyright, and Competition*, 195 (1989); *Conference Documents, LaST Frontier Conference Report on Copyright Protection of Computer Software*, 30 Jurimetrics J. 18 (1989).

[120]17 U.S.C. §101 (1988) (defining "computer program").

[121]17 U.S.C §101 (1988); H.R. Rep. 94-1476 at 54, *reprinted in* 1976 U.S. Code Cong. & Admin. News 5667 (computer programs are literary works under the copyright statutes); *LaST Frontier Conference Report on Copyright Protection of Computer Software*, 30 Jurimetrics J. 18 (1989).

[122]Goldberg and Furleigh, *supra*, note 97 at 160.

[123]Idem at 171.

[124]740 F. Supp. at 68.

[125]Goldberg and Furleigh, *supra*, note 97 at 158; *LaST Frontier Conference*, *supra*, note 99 at 18.

[126]H.R. Rep. No. 1476, 94th Cong., 2d Sess. 54, *reprinted in* 1976 U.S. Code Cong. & Admin. News 5659, 5667.

as books, periodicals, manuscripts, phonorecords, film, tapes, disks, or cards, in which they are embodied."[127] A consequence of considering software a literary work rather than one of the other six copyright classes or a new class is that the legal precedent applicable to literary works applies to software. Literary works such as novels or plays may be infringed by copying their literal words or by copying their nonliteral plot or characters.[128]

5.2 The Software Development "Clean-Room" Procedure

The software development clean-room procedure permits copying the idea of a copyrighted expression while avoiding infringement of its literal source expression.[129] One group of computer scientists studies the copyrighted code and writes functional specifications for a separate group of computer scientists who will write a new program using those functional specifications. The clean-room procedure avoids infringement of software's literal aspects, because the computer scientists who write the new program have never seen the copyrighted source. The functional specifications they receive from the computer scientists who read the source code contain only descriptions of what the code does but do not contain any code.[130]

The copyrighted work is not infringed, because the idea is not copyrightable and because independent creation of a similar or even identical expression (source code) from the idea is not infringement.[131]

---

[127]17 U.S.C. §101 (1988).

[128]*See Nichols v. Universal Pictures Corp.*, 45 F.2d 119 (2d Cir. 1930).

[129]See Derwin, *Licensing Software Created Under "Clean Room" Conditions*, 2 Computer Software 1989 Protection and Marketing 439 (1989); Clapes, *supra*, note 110 at 153 ("'Clean room' procedures are procedures for isolating persons who are developing software intended to be competitive with existing software offered by others, and for maintaining records that will establish that their work was done without copying.").

[130]See Hayes, *supra*, note 123 at 18-20.

[131]*Herbert Rosenthal Jewelry Corp. v. Kalpakian*, 446 F.2d 738, 741 (9th Cir. 1971); Hayes, *Acquiring and Protecting Technology: The Intellectual Property Audit*, 8 Computer Lawyer 18 (April 1991); Clapes, *supra*, note 104 at 152-153 ("One who, without copying, produces a similar or even identical text has not violated the copyright owner's exclusive right.").

5.3 Why the Clean-Room Procedure Cannot Guarantee Protection Against Nonliteral Software Infringement

There is no infringement if the functional specifications provided to the computer scientists who write the new program do not contain any literal or nonliteral aspects of the copyrighted program. A line-by-line comparison of the code and functional specifications can verify that no copyrighted literal code has been passed to the writers of the new program. No similar procedure can guarantee that the functional specifications contain no copyrighted nonliteral aspects.

5.4 Copyright Infringement Risks for the Software Developer

All software developers, private or government, who have seen copyrighted software risk copyright infringement when they write or modify software. The federal government has waived sovereign immunity and consented to suit for copyright infringement.[132] In 1990, Congress waived state government sovereign immunity from copyright infringement suits.[133] Any copyrighted software viewed by the developer could be infringed when the developer writes or modifies code if the copyrighted work is intentionally or unintentionally reproduced in the new code.

Unintentional or subconscious copying of a copyrighted work can constitute copyright infringement; infringement may arise from "subconscious memory derived from hearing, seeing or reading the copyrighted work at some time in the past."[134] The risk of unintentional or subconscious infringement of a user interface's nonliteral aspects is particularly high. By design, the user interface is the most visible aspect of software. User

---

[132]28 U.S.C. §1498(b) (1988). (This sovereign immunity waiver applies to infringements "by the United States, by a corporation owned or controlled by the United States, or by a contractor, subcontractor, or any person, firm, or corporation acting for the Government and with the authorization or consent of the government. . ."); (Kwall, Governmental Use of Copyrighted Property: The Sovereign's Prerogative, 67 TEX. L. REV. 685, 753 [1989]).

[133]Copyright Remedy Clarification Act, Pub. L. No. 101-553, 104 Stat. 2749 (1990) ("any State, any instrumentality of a State, and any officer of employee of a State or instrumentality of a State acting in his or her official capacity, shall not be immune, under the Eleventh Amendment. . .or under any other doctrine of sovereign immunity, from suit in Federal Court. . .").

[134]*Herbert Rosenthal Jewelry Corp. v. Kalpakian*, 446 F.2d 738, 741 (9th Cir. 1971) (*citing Howell's Copyright Law* 129 [4th ed. 1962]); *Buck v. Jewell-La Salle Realty Co.*, 283 U.S. 191, 198 ("Yet unconscious plagiarism is actionable quite as much as deliberate."). See also R. Brown and R. Denicola, *Cases on Copyright* 435 (5th ed. 1990); Raysman and Brown, *"Lotus" Decision: Greater Protection for Computer Programs*, New York Law Journal (August 14, 1990).

interfaces are frequently displayed on television, in publications, in computer stores, at computer trade shows, and in offices. The literal code, however, is rarely visible.

My employer, an executive branch department of the United States government, has repeatedly warned its employees and officers that intentional verbatim duplication of copyrighted works is illegal. In fifteen years as a computer scientist, however, I have never been warned by my employer that copyright infringement might result from copying nonliteral aspects, intentionally or unintentionally.

Given the proliferation of personal computers in the office and the trend toward user configurable user interfaces, it is imperative that all computer users, not just computer scientists, are warned that copying literal or nonliteral aspects of copyrighted programs may be copyright infringement.

5.5 Conclusion

User interface compatibility is a complex issue involving the user and the similarity, friendliness, and consistency of interfaces. Although identical interfaces are compatible for all users, compatible interfaces need not be identical. Experienced users can readily adapt to some dissimilarities, while novice users use only a subset of the available functions and thus need compatibility only for those few functions.

According to the *Lotus* court, some elements of user interfaces are copyrightable subject matter. Infringement occurs when a user interface copies enough of these elements to become substantially similar to the copyrighted interface. Neither the boundary between copyrightable subject matter and noncopyrightable subject matter (the idea-expression boundary) nor the boundary between permissible copying and copyright infringement (substantial similarity) is well defined.

The difficulty in identifying copyrighted user interfaces elements and the ease and frequency of viewing them significantly increase the risk of unintentional copyright infringement for anyone who writes or modifies software and for their employers.

Bibliography

1. James Martin, *Design of Man-Computer Dialogues* (Prentice-Hall: 1973).

2. Roger Ehrich and Robert Williges, Eds., *2 Human-Computer Dialogue Design* (Elsevier: 1986).

3. Henry Simpson, *Design of User-Friendly Programs for Small Computers* (McGraw-Hill: 1985).

4. K. Hopper and I.A. Newman, Eds., *Foundation for Human-Computer Communication* (1986).

5. Albert Badre and Ben Shneiderman, Eds., *Directions in Human/Computer Interaction* (Ablex: 1982).

6. Danny Kopec and Donald Michie, *Mismatch Between Machine Representations and Human Concepts, Official Publications of European Communities* (1983).

7. Lorraine Borman and Bill Curtis, Eds., *Human Factors in Computing Systems* (ACM: 1985).

8. B. Christie, Ed., *Human Factors of the User-System Interface* (North-Holland: 1985).

9. Thomas Bernold, Ed., *User Interfaces: Gateway or Bottleneck?* (North-Holland: 1988).

10. Yannis Vassiliou, Ed., *Human Factors and Interactive Computer Systems* (Ablex: 1984).

11. Anthony Clapes, *Software, Copyright, and Competition: The "Look and Feel" of the Law* (Quorum: 1989).

12. John Lawrence and Bernard Timberg, Eds., *Fair Use and Free Inquiry: Copyright Law and the New Media* (Ablex: 1989, 2nd ed.).

13. Morton Goldberg [Chairman PLI], *Computer Software 1989 Protection & Marketing*, vols. 1 & 2 (Practicing Law Institute: 1989).

14. Morton Goldberg [Chairman PLI], *Computer Software 1990* (Practicing Law Institute: 1990).

15. Ralph Brown and Robert Denicola, *Cases on Copyright, Unfair Competition, and Other Topics Bearing on the Protection of Literary, Musical, and Artistic Works* Foundation Press: 1990, 5th ed.).

16. Jon Baumgarten and E. Gabriel Perle [Co-Chairmen], *The U.S. Copyright Office Speaks* (Prentice-Hall: 1989).

PPRAM

ISBN 1-879716-04-6