TAILORING LEGAL PROTECTION FOR COMPUTER SOFTWARE

Peter S. Menell*

Discussion Paper No. 26

4/87

Program in Law and Economics
Harvard Law School
Cambridge, MA 02138

The Program in Law and Economics is supported by a grant from the John M. Olin Foundation.

*Mr. Menell, a 1986 graduate of Harvard Law School, is currently law clerk to Hon. Jon O. Newman, and will become Assistant Professor of Law at Georgetown University Law Center in the coming academic year.

forthcoming in
39 Stanford Law Review (July 1987)

TAILORING LEGAL PROTECTION FOR COMPUTER SOFTWARE

Peter S. Menell

forthcoming in 39 Stanford Law Review (July 1987)

Tailoring Legal Protection for Computer Software

Peter S. Menell

Abstract

This paper presents an economic analysis of legal protection for computer software. The unique technological and economic attributes of computers and computer markets significantly alter the traditional public goods analysis for legal protection. Because of the many interrelated stages of the computer indsutry and the high costs of contracting among the diverse producers and consumers, expansive legal protection for software threatens innovation and dissemination of new, improved, and less expensive products. In addition, the network externality associated with computer operating systems--commonly referred to as the problem of compatibility--suggests that braod legal protection for computer software discourages adoption of uniform standards and has implications for incentives to innovate new operating systems. The paper argues that a hybrid patent/compulsory licensing scheme for operating systems would better promote innovation and dissemination of improved products and realization of network benefits. The paper also suggests ways in which copyright protection for application programs could be better tailored to achieve the objectives of legal protection.

TAILORING LEGAL PROTECTION FOR COMPUTER SOFTWARE

Table of Contents

Introduction

It became evident by the mid-1970s that intellectual work embodied in new technologies -- in particular, computer software -- did not fit neatly within the traditional forms of legal protection for intellectual property.[1] Because of the complexity of these issues, Congress created the National Commission on New Technological Uses of Copyrighted Works (CONTU) for the purpose of recommending a national policy that would provide adequate legal protection for intellectual work embodied in new technologies while ensuring public access.[2] The Commission conducted hearings and received expert reports beginning in May, 1976. On July 31, 1978, CONTU recommended that full copyright protection be extended to all forms of computer software.[3] Congress implemented this recommendation in the Federal Copyright Act of 1980.[4]

The computer industry has undergone dramatic change in the eight years since CONTU issued its recommendations.[5] In 1978, there were fewer than 200,000 microcomputers (costing more than $1000) in the United States.[6] By the end of 1986, there are estimated to have been more than 12 million.[7] Over this same period, computer software revenues will have grown at an annual rate of more than 20%.[8]

Problems that have emerged during this period of growth -- impediments to entry by small, independent firms,[9] wasteful expenditure of vast resources to develop non-infringing, yet compatible computer systems,[10] and the lack of clear industry standards[11] -- have highlighted serious shortcomings of CONTU's analysis and recommendations. These problems can be traced to CONTU's inadequate appreciation of the public goods problem raised by computer software and CONTU's failure to recognize the network externality inherent in computer systems.

The members of CONTU recognized that authors and inventors might not be able to appropriate an adequate reward for their effort when the cost of copying software is low.[12] CONTU naively assumed that simply affording broad protection to all forms of computer software would encourage the optimal level of innovation in computer technology, thereby promoting the public interest. What CONTU failed to appreciate, however, are the potentially detrimental effects of bestowing broad legal protection at a critical level of a complex industry.

What ultimately determines the social value of legal protection for intellectual property is the speed at and extent to which it fosters the availability of new, improved, and less expensive products. Technological innovations at particular stages of the industry are important determinants of social welfare; but only to the extent they are diffused and adopted. Given the many interrelated stages of the computer industry -- basic research, operating systems, hardware products, integrated hardware systems, peripherals, application programs, marketing -- and the high costs of contracting among the diverse producers and consumers at each stage, expansive legal protection at an early stage inhibits innovation at the other stages, thereby slowing diffusion and adoption. In addition, when technological advancement at one stage proceeds by enhancing prior innovations, bundling innovations, and applying prior innovations in a new area -- as in applications programming -- long-lived protection slows innovation, diffusion, and adoption.

Moreover, CONTU failed to recognize the importance of the fact that computers are a systems technology.[13] Uniform product standards broaden the availability of complementary products such as application programs. Affording full copyright protection to operating systems -- the source of compatibility -- can discourage adoption of widespread product standards and technological innovation.

This Article addresses these shortcomings of the CONTU recommendations by presenting an economic analysis of legal protection for computer software.[14] Before we can undertake such an analysis, it is important to have some familiarity with the technological, economic, and legal landscape. Part I describes salient features of computer technology and Part II discusses the markets for computers and computer products. Part III then identifies and describes the two primary sources of market failure that might justify government intervention -- the public goods nature of information and the network externality flowing from industry-wide standards. Part IV describes the current state of legal protection for computer software in the United States.

Parts V and VI analyze, respectively, legal protection for computer operating systems and application programs. They first assess the need for legal protection. They then analyze the economic effects of using copyright doctrine to protect these forms of computer software. These Parts conclude by outlining modes of legal protection that are better tailored to remedying the market failures in the computer software market. Part VII offers some concluding remarks.

I. Computer Technology

An understanding of the basic technology of computers is essential to analyzing legal protection for computer software. This Part briefly reviews the salient features of computer technology. Section A discusses the design of computers, highlighting the relationship between hardware and software. Section B discusses the major types of computer software.

A. Overview of Computer Processing

1. Evolution of the Modern Computer

The first computing machines were comprised completely of hardware. These "dedicated" counting machines could perform only the one function that they were wired to perform. They had to be rewired in order to perform a different function.

During the 1940's, scientists developed the first machines that could store and use encoded instructions or programs. The central processing unit (CPU) is the actual computer in these programmable or "universal" machines. The CPU is wired to perform a basic set of "primitive functions" such as addition and multiplication. In essence, a computer processes data by performing controlled sequences of these primitive functions.

The flexibility provided by programmability greatly enhanced the utility of computers. Limitations of electronic technology, however, constrained the computing power of the first generation of computers.[I.1] The electronic vacuum tubes that ran these computers were bulky and consumed large amounts of energy. The invention of the transistor in the late 1940s greatly expanded the capability of computers. Since information in computers is stored and processed using binary electronic switches, the great challenge for the computer industry was to reduce the size and compactness of these switches. The development of integrated circuits enabled many switches to be fit within thin layers of semiconductor material. By the 1970s, "semiconductor chips" containing more than 100,000 transistors were being used in computers.

As chip technology advanced, the size of computers decreased while their computing power increased. The early computers were predominantly large and expensive "mainframes". By the early 1960s, advances in electronics technology enabled computer firms to manufacture minicomputers. Further advances in electronics -- in particular, the development of very large-scale integrated circuits -- have made micro- or personal computers possible.

2. Modern Computer Design

The basic hardware of a modern computer system includes a CPU, internal memory storage, and disk drives or other devices for transfering data and programs into and out of the internal memory. The internal memory typically features two types of chips: random access memory (RAM) and read-only memory (ROM). RAM chips serve primarily as temporary storage devices, though they can also serve as permanent memory for data or programs. Data can be input into RAMs, erased, or altered. ROM chips have memory permanently embedded in them and therefore can only be read by the computer. This information cannot be altered by the computer system.

Computer engineers design the programming capability of a computer to suit the user's needs. By building more of the desired function directly into the wiring of the computer, they can achieve more efficient processing. This greater speed, however, comes at a cost of less flexibility -- that is, less ability to run a wide range of programs. This technological trade-off harks back to the early days of computer technology when all programs were wired into the computer. Advances in computer technology have made greater efficiencies of processing possible without the need to hard-wire the computer. When a user has only a few computing needs or when high speed processing is desired, however, it may still better to rely heavily upon internal programming.

Most microcomputers feature a high degree of programming flexibility. These computers typically have a general operating system that controls the hardware components of the system. The operating system also assists the computer in reading and implementing externally stored application programs. The range of application programs that can be run on a computer determines the computer's flexibility. The next section will describe the nature of computer software in more detail.

B. Types of Computer Software

Computer programs can be written in object code or source code. Object code is sequences of binary units (0 or 1) that can be read directly by the computer -- in essence, a binary unit simply indicates whether particular electronic switches should be in the on or off position. Source code is sequences of instructions in higher level computer language (such as Fortran or PL-1). Programs written in source code must be translated into object code in order to be read by the computer. This is accomplished by compilers (or translaters) within the computer.

There exists a complex hierarchy of computer programs. The two major categories of programs are: (1) operating systems; and (2) application programs.[I.2] Operating systems manage the internal functions of the computer.[I.3] They coordinate the reading and writing of data between the internal memory and the external devices (e.g., disk drives, keyboard, printer), perform basic housekeeping functions for the computer system, and facilitate use of application programs. In essence, the operating system prepares the computer to execute the application programs.

Application programs perform the wide range of data processing tasks sought by the computer user. Examples of application programs include include bookkeeping programs, word processing programs, data processing programs, and video games.

II. The Markets for Computers and Computer Products

At the most basic level, consumers demand "computing services" to meet their data processing needs. As we saw in Part I, these needs can be satisfied completely by hardware;

or they can be satisified by a general purpose computer equipped with the appropriate application software. Thus, hardware and software are both substitute and complementary commodities. Consequently, although this Article focuses on legal protection for computer software, it will be important to consider the implications of modes of legal protection for the broader markets of computers and computer products.

A. Demand for Computer Services

The demand for computer services is driven by the great diversity of entities -- businesses, government agencies, research institutions, individuals -- with data processing needs. These needs range from simple calculations to complex scientific applications. Consumers also differ in the variety of data processing tasks that they must accomplish. A medium-sized business, for example, might have many data processing tasks for which a computer might prove useful: handling the payroll, record-keeping, word processing, and projecting business trends. In contrast, a manufacturing company might simply need to regulate the temperature of a kiln. A physicist might need a computer to execute high speed calculations.

B. Supply of Computer Services

The hardware sector of the computer industry consists of original equipment manufacturers (OEMs), semiconductor chip manufacturers, and vendors. The vendors purchase computer components and chips from the other firms and assemble them into computer systems. A few dominant firms in the industry like IBM are involved in all aspects of hardware research and development.

The software sector offers a wide variety of services and products. This work includes the design of general operating systems, contract programming, and the development of commerical application packages. Many large hardware systems manufacturers develop operating and application software for their systems. There are also many small, independent firms specializing in aspects of software services and product development.

C. Evolution of the Computer Industry

The rapid advancement of all aspects of computer technology has enabled the computer services market to expand at a blistering pace since the advent of commercial computing in the mid 1950s. Since 1955, machine performance has increased by 6 orders of magnitude.[II.1] The greater sophistication, enhanced flexibility, and lower cost of computing power has greatly expanded the market for computing services. Computers -- which were not long ago found only at large corporations, research institutions, and government agencies -- are now in a substantial portion of American homes.[II.2]

In the hardware field, the trend has been toward smaller, universal computer systems. In 1965, domestic consumers purchased 260 minicomputers and 5,350 mainframes.[II.3] Minicomputer unit sales surpassed mainframe unit sales by 1974. And microcomputer unit sales surpassed minicomputer unit sales in their second year of production (1976). It is estimated that sales of microcomputers (costing more than $1000) reached approximately four million units for 1986. This translates into revenues of almost $12 billion, giving microcomputers the largest share of computer revenues.[II.4]

These trends in hardware have dramatically changed the structure of the software sector of the industry. Just 10 years ago, most software firms produced custom programs for the predominantly commerical customers.[II.5] The advent and proliferation of microcomputers and the increase in flexibility of minicomputers has greatly increased the demand for general purpose application packages. Consequently, firms producing commercial application packages for the variety of computer systems have emerged as the major revenue generating force in the computer industry.[II.6]

Software is expected to play an increasingly important role in the computer market. The Computer and Business Equipment Manufacturers Association predicts that software revenues will grow at an annual rate of 13.6% through 1990, compared to a rate of 9.7% for the hardware sector of the industry.[II.7] In the coming years, a major focus of the industry will be toward connecting the vast array of computer and telecommunication systems.[II.8]

III. <u>Economic Justifications for Government Intervention in the Market for Computer Software</u>

A fundamental reference point for economic analysis is the proposition that in the absence of market imperfections,[III.1] competition will assure an efficient allocation of resources.[III.2] This Part provides a general discussion of two market failures prevalent in the market for computer software. Section A discusses the market failure endemic to all markets in goods that embody technological innovations -- the public goods aspect of information. Section B discusses the market failure created when individuals' utility or satisfaction from consuming a good depends on the number of other persons consuming the same good.[III.3] This might arise in the computer field, for example, because the availability of application programs for use with a particular operating system depends upon the number of people owning microcomputers featuring that operating system. Section C explains why these market failures and the technology and structure of the software industry suggest that a functional distinction be made in analyzing legal protection for operating systems and application programs.

A. Innovation as a Public Good

All markets for goods embodying intellectual work exhibit an externality commonly referred to as the "public goods" problem.[III.4] Public goods have two distinguishing features: (1) non-excludibility -- it is difficult to exclude those who do not pay for a good from consuming it; and (2) non-rivalrous competition -- additional consumers of the good do not deplete the supply of the good available to others. Beautiful gardens and military defense are classic examples of public goods. The private market will undersupply these goods because producers cannot reap the marginal value of their investment in providing such goods.[III.5]

As CONTU well recognized, the information comprising innovations in computer software is a prime example of a public good.[III.6] Given the availability of low-cost copying technologies, it is often impossible to exclude nonpurchasers from the benefits of innovations embodied in computer programs once they are made commercially available. Moreover, one person's use of the information does not detract from any other person's use of that same information. Since the authors and creators of computer software cannot reap the marginal value of their efforts, they will, in the absence of other incentives to innovate, undersupply technological advances in computer software. The government typically alleviates the public goods problem in generating innovation and original expression by bestowing limited legal protection for goods embodying novel ideas and literary works containing original expression.

1. The Nature of Laws Protecting Intellectual Property

The basic linkage between the scope of intellectual property protection and the public welfare involves three steps.[III.7] First, enhancing the scope of intellectual property protection (e.g., by increasing the term of legal protection or expanding the breadth of legal protection) increases the expected reward to the creator by enhancing the opportunity for monopolistic exploitation of any works created. Second, increased rewards encourage inventive activity. Moreover, the disclosure of new discoveries that is fostered by protection further spurs inventive activity. Third, greater investment in inventive activity results in the discovery of more ideas and faster advancement of technology, thereby increasing the range of products and reducing the costs of products to society.

This basic linkage, however, is greatly complicated in markets for products in which innovation occurs at many stages. What ultimately determines the social value of technological progress is the speed at and extent to which new, improved, and less expensive products are available. Individual technological innovations at particular intermediate stages are important determinants; but the pattern of adoption and diffusion of these innovations is of critical importance.[III.8] Historical and industry studies of the innovation process find that inventions are highly interdependent; "technologies undergo a gradual, evolutionary development which is intimately bound up with the course of their diffusion."[III.9] In fact, "secondary inventions" -- including essential design improvements, refinements, and adaptations to a variety of uses -- are often as crucial to the generation of social benefits as the initial discovery.[III.10]

These interactions have been and continue to be particularly important in the evolution of computer technology. Advances in computer technology are made at many interrelated levels -- basic research, system-unit hardware, operating systems, peripheral equipment hardware, application programming, marketing -- by diverse individuals, firms, and research institutions. It cannot be assumed automatically, therefore, that expansive legal protection for intellectual property at any one level will generate both the optimal amount of innovation and the optimal diffusion path.[III.12]

2. Assessing the Need for Legal Protection for Intellectual Work

The failure of one of the assumptions underlying the efficiency of the free market system is only a necessary condition for government intervention; it is not a sufficient condition. Because of the costs of legal protection (in terms of adverse effects on the market directly affected as well as distortions in other markets), the unregulated market, though not efficient, might still perform better than government regulation.[III.13] Thus, it is important to assess the extent to which other forces -- both market and non-market -- tend to offset the adverse effects of the public goods problem.

The market itself often provides means by which producers of public goods can realize sufficient rewards to encourage them to produce such goods. Often being the first firm to introduce a product earns the innovator substantial and long-lived advantages.[III.14] In addition, producers of products embodying innovations can internalize some of the benefits of their research efforts by requiring purchasers to enter into long-term maintenance and updating contracts. They can also require purchasers to enter into licensing agreements prohibiting reproduction of the product and dissemination of information embodied in the product.

Alternatively, producers can utilize technological means for preventing those who do not pay for the good from enjoying the benefits. For example, anticopying devices can impede reproduction and disclosure of intellectual work embodied in products. If these means of protecting research and development are inexpensive and effective, then legal protection may not be needed to ensure efficient provision of the good. Indeed, there would be no appreciable public goods problem.

Government and private subsidies of research and development can also alleviate the public goods problem. Government research and development subsidies, particularly through the Department of Defense, have been extremely important in the development of computer technology.[III.15] Moreover, universities, whose work product typically is in the public domain, have played an important role in the development of computer technology.

3. The Design of Legal Protection for Intellectual Work: Balancing Costs and Benefits

As the discussion of diffusion issues above indicates, legal protection for intellectual property engenders real costs to society. In addition to the direct costs of research and development associated with increased inventive activity, enlarging the scope of intellectual property protection increases the losses due to monopolistic exploitation of innovations. These losses are particularly regrettable for innovations that would have been created in the absence of legal protection. Moreover, a broad regime of intellectual property protection might inhibit inventive activity by competitors and producers of complementary and downstream products. Finally, a system of protection of intellectual property will entail administrative expenses, including the costs of keeping abreast of the legal rights of others and enforcing legal rights.

From the perspective of the public interest, the optimal system for protecting intellectual work equates the marginal benefit of enhancing the scope of intellectual work protection -- in terms of the availability of more and better products -- with the marginal cost of greater protection -- in terms of research costs, losses due to monopoly exploitation, administrative costs, and inhibiting effects on inventive activity.

B. <u>Network Externalities</u>

The second principal market failure in the computer software market, one which CONTU overlooked in its analysis and recommendations, arises from the presence of network externalities. Network externalities exist in markets for products for which the utility or satisfaction that a consumer derives from the product increases with the number of other consumers of the product.[III.16] The telephone is a classic example of a product for which there are network externalities. The benefits to a person from owning a telephone are a function of the number of other people owning telephones connected to the same telephone network; the more people on the network, the more people each person can call and receive calls from. Another classic network externality flows from the prevalence of a standard type-writer keyboard.[III.17] Because almost all English language typewriters feature the same keyboard configuration, commonly referred to as "QWERTY", typists need learn only one keyboard system.

Network externalities also inhere to product standards which allow for the interchangeability of complementary products.[III.18] Examples of products for which this type of network externality is important are video cassette recorders, phonographs, and computer operating systems for microcomputers.[III.19] As we discussed in Part I, general computer systems have developed that allow consumers to use a variety of application software programs on the same system-unit hardware. The only requirement is that the application program be coded to work on the operating system embedded in the general computer system. Thus, the operating system serves as a compatibility nexus for a particular computer network. Application software producers will develop more programs for systems that are widely used; hardware producers will develop more configurations of disk drives, memory, and other features for popular operating systems. In general, the benefits of a larger computer operating system network include a wider variety of application software that can run on the operating system, lower search costs for consumers seeking particular application programs that run on the operating system, and wider availability of compatible hardware configurations and peripherals.

1. The Effect of Legal Protection for Product Standards on the Realization of Network Externalities

An important economic consideration in markets featuring network externalities is whether firms will have the correct incentives to adopt compatible products, thereby enlarging existing networks. Professors Katz and Shapiro demonstrate that firms with brand recognition might prefer to adopt non-compatible product standards even though net social welfare would be improved by their adoption of compatible products.[III.20] The explanation for this behavior is that by adopting a compatible standard, a firm enlarges the size of a network that comprises both the adopter's product and those of its rivals. This will have the effect of increasing the desirability of rivals' products to consumers, thereby reducing the adopter's market share relative to what it might be had the firm adopted a non-compatible product standard.[III.21]

The availability of legal protection for product standards strengthens this adverse incentive by increasing the rewards that firms with brand recognition can reap by increasing the rewards to developing non-compatible products standards. In the absence of legal protection for product standards, the private benefits from introducing a non-compatible product standard will be short-lived. As the product standard gains acceptance in the marketplace, other firms -- perhaps those without wide brand recognition -- will adopt the new product standard, thereby competing away the market share of the first producer. By contrast, the availability of legal protection for product standards greatly increases the rewards that a firm can reap by successfully introducing a non-compatible product standard. It allows a firm to enter a market without expanding the network of its rivals, while enabling the firm to obtain the exclusive right to manufacture and sell products embodying its standard. In this way, the firm can enjoy a long-term monopoly in the standard, with the option, of course, of licensing the standard to others at a significant royalty.

Should a proprietary product standard become a <u>de facto</u> industry standard, the magnitude of external benefits from the network will depend on the ability of the "dominant" firm to serve the market (i.e., through diffusion of the products) and the transaction costs of licensing the standard to firms that can better serve segments of the market.[III.22] Thus when consumers have homogeneous demands, one firm with mass-production and marketing capabilities can adequately serve the market, ensuring that the full potential benefits of the network are realized. But when consumers have heterogenous demands -- i.e., their needs are sufficiently specialized that one manufacturing and marketing organization cannot adequately serve all of them -- and costs of licensing are high, the network will not expand sufficiently to generate its full potential of external benefits.

2. <u>The Effects of Legal Protection for Product Standards on Innovation in and Adoption of New Product Standards</u>

While a widely adopted product standard can offer important benefits to consumers and firms, it can also "trap" the industry in an obsolete or inferior standard.[III.23] In essence, the installed base built upon the "old" standard -- reflected in durable goods and human capital (training) specific to the old standard -- can create an inertia making it that much more difficult for any one producer to break away from the old standard by introducing a non-compatible product, even if the "new" standard offers a significant technological improvement over the current standard.[III.24] In this way, network externalities can retard innovation and slow or prevent adoption of improved product standards.[III.25]

As an example of this phenomenon, investigators cite the persistence of the standard "QWERTY" typewriter keyboard despite the availability of a better key configuation developed by August Dvorak and W.L. Dealey in 1932.[III.26] Adoption of the better standard appears to have been effectively stymied by switching costs -- the costs of converting or replacing "QWERTY" keyboards and retraining those who use this system. Because of the fear that national standards would exacerbate the excess inertia problem, the National Bureau of Standards declined to set interface standards for computers in the early 1970's.[III.27]

The availability of the proper mode of legal protection for product standards can alleviate this excess inertia effect by assuring innovators of better standards a limited monopoly in the event their standards break into the market. Without the availability of legal protection, innovators' profits would be competed away as other firms introduced competing products embodying the better standard. It should be noted in this regard that legal protection is more important for encouraging innovation in product standards by smaller firms than for firms with well-established reputations because the latter have less difficulty in establishing new product standards.

At the same time, however, affording legal protection for product standards encourages investment by competing firms in efforts to circumvent legal protection. Firms that are prevented from using established standards often invest substantial resources in attempts to develop compatible, though non-infringing, products. From the broader social perspective, investment in research resources solely to circumvent legal protection, as opposed to advancing the state of the art, is wasteful.

3. The Effect of Legal Protection for Product Standards on Competition and Innovation in Complementary Products

In many cases, a product standard is but one component of the ultimate product. In the computer field, for example, operating systems (and related ROM chips) are typically sold as part of computer packages comprising a basic system-unit, software, and peripherals. By offering the component embodying a product standard only as part of a package of components, a firm that enjoys legal protection for the product standard can effectively prevent other firms from offering products that utilize the product standard.

In a static model of competition, there is little reason to fear that this tying phenomenon [III.28] will be any more harmful from an efficiency standpoint than the general anticompetitive effect of monopoly power in the proprietary product standard.[III.30] All monopoly profits that can be obtained from exploitation of packages comprising the protected product standard can be reaped by exploitation of the product standard component alone. When dynamic considerations are incorporated into the analysis, particularly effects on innovation and maintenance of market power over time, tying can add to the social welfare losses associated with static monopoly power.[III.31] By tying a product standard to complementary products, a monopolist can effectively discourage other firms from attempting to improve such complementary products because there might be little or no market for such products. This can delay socially valuable innovations and prolong the existence of monopoly power.

Even in the presence of these dynamic considerations, a firm will not necessarily market its proprietary product standard in only the most complete ultimate products. It will want its product line to appeal to a wide range of consumers -- from those seeking a simple version of the product to those seeking all of the extras. Moreover, by limiting the variety in which its product standard is available, the firm enhances the appeal of rivals' product standards, particularly those that come in a range of models. These factors affect the extent to which a firm which possesses a proprietary product standard will allow competitors to offer products that complement its product standard.

4. Government Policies to Address Network Externalities

Network externalities present a problem of coordination among decentralized agents.[III.32] The government has numerous means by which to promote coordination. It can set standards directly, relax antitrust restraints that prevent private firms from setting voluntary standards,[III.33] promote standardization through its market power as a major purchaser,[III.34] and facilitate access to industry standards through the design of laws protecting intellectual property. In other words, the question is the proper mode by which the government can strengthen sanctioning of sponsorship of standards to promote an optimal balance of competition, realization of network externalities, and innovation.

In this Article, we are concerned primarily with the role of legal protection as a means of accomplishing these objectives. As we will see below, however, legal protection can play a detrimental role if not properly tailored to market conditions. In designing legal protection for intellectual work featuring network externalities, policymakers should foster the realization of benefits from standardization -- wider availability of products, lower search costs, vigorous competition -- while minimizing the potentially adverse effects of universal access to new technologies on incentives to innovate better product standards.

C. A Functional Distinction Between Operating Systems and Application Programs

Every computer program invokes different degrees of the public goods and network externalities market failures discussed above. Before embarking on formal analysis of legal protection for computer software, therefore, it is necessary to divide up the product space into useful categories of study. If a category is too broad, then the analysis may overlook important distinctions among products. On the other hand, if the product space is divided into too many categories, there is a risk that policy recommendations will be too ad hoc to be administratively feasible.[III.35]

Fortunately, a fairly clear demarcation emerges from our review of technological and economic aspects of computer software and the principal market failures affecting its provision. Our discussions of technological aspects of types of computer software and network externalities highlight a critical distinction between operating systems and application programs. Operating systems establish standard protocols and formats to which application programs and some peripheral equipment must be tailored. Application programs, by contrast, primarily access the computer to perform specific user tasks. They do not in general serve as a standard for other software or hardware. Because the network externality concerns associated with operating systems are fundamentally different from those associated with application programs, it seems sensible, at least as a first cut, to analyze legal protection for these two categories of computer software separately.[III.36]

It will be important to keep in mind, however, that the distinction between these two categories of software products, like other distinctions in the computer field,[III.37] is not crystal clear and is subject to change as technology advances.[III.38] Moreover, there is risk that legal distinctions, by altering the nature of property rights, will encourage those affected to develop products that provide maximum legal protection rather than social benefit. Despite these cautionary notes, there are clear economic principles that could guide regulatory authorities and courts in interpreting the distinction. In particular, the distinction would turn largely on whether the product in question serves as a standard that affects access to an important market.[III.39]

In the next Part, we present an overview of the various forms of legal protection currently available for computer software. Parts V and VI then analyze legal protection for operating systems and application programs, respectively.

IV. Legal Protection for Computer Software in the United States

Creators of computer software may seek legal protection under the three traditional forms of legal protection for intellectual property: copyright, patent, and trade secret law. In addition, under a 1984 law, producers of semiconductor chips can now protect the intellectual work embodied in such chips under a hybrid form of legal protection. Of these forms of legal protection, copyright typically provides the most direct and easily attainable protection for computer software.

A. Copyright Law

Under copyright law,[IV.1] a work must satisfy two principal requirements in order to obtain copyright protection: (1) it must be an "original wor[k] of authorship"; and (2) it must be "fixed in [a] tangible medium of expression."[IV.2] An author of a copyrightable work receives exclusive rights to the use of that work for the author's life plus fifty years.[IV.3] Copyright law protects the form in which an idea appears rather than the idea itself.[IV.3]

The copyright statute enumerates seven categories under "works of authorship" including "literary works", defined as follows:

> 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 as books, periodicals, manuscripts, phonorecords, film, tapes, disks, or cards, in which they are embodied.[IV.4]

A work is "fixed" in a tangible medium of expression when:

> its embodiment in a copy or phonorecord, by or under the authority of the author, is sufficiently permanent or stable to permit it to be perceived, reproduced, or otherwise communicated for a period of more than transitory duration. A work consisting of sounds, images, or both, that are being transmitted, is "fixed" for purposes of this title if a fixation of the work is being made simultaneously with its transmission.[IV.5]

Although section 102(a) does not expressly list computer programs as works of authorship, the legislative history suggests that Congress considered programs to be copyrightable as literary works.[IV.6] Any ambiguity about whether copyright protected computer programs was resolved in 1980 when Congress amended the copyright law by adding a definition of a computer program: "A 'computer program' is a set of statements or instructions to be used directly or indirectly in a computer in order to bring about a certain result."[IV.7] The amendments also substituted a new section 117 which provides that "it is not an infringement for the owner of a computer program to make or authorize the making of another copy or adaptation of that computer program" when necessary to "the utilization of the computer program" or "for archival purposes only."[IV.8] The language of this section, by carving out an exception to the normal proscriptions against copying, clearly indicates that programs are copyrightable and are otherwise afforded copyright protection.

The scope of these provisions have since been delineated by the courts. In a landmark case, Apple Computer v. Franklin Computer,[IV.9] the Third Circuit held that the Copyright Act extends to operating programs as well as application programs, whether fixed in source code or object code or embodied in read-only-memory (ROM).[IV.10]

B. Patent Law

Patent law provides protection to "any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof . . . ."[IV.11] A patent protects its owner not only from competition by copiers of the inventor's work, but also from those who independently discover the advance.[IV.12] Recipients of patent protection enjoy rights to exclusive use of the subject matter of their intellectual work for 17 years.

Unlike copyright law, patent law protects the utilitarian aspects of a work and not just the particular way the invention is expressed in the patent application.[IV.13] So as not to unduly hinder scientific and technological progress, however, the subject matter of patent has traditionally been restricted so as not to allow an inventor to obtain protection solely for mental processes[IV.14], scientific principles [IV.15], laws of nature [IV.16], or mathematical algorithms.[IV.17]

The Patent and Trademark Office initially took the view that computer programs were not patentable subject matter because they could be characterized as sequences of mental steps and/or mathematical algorithms.[IV.18] In the first computer program related cases to reach it, the Court of Customs and Patent Appeals ("CCPA")[IV.19] took a more favorable view toward the patentability of computer programs. Rejecting the conclusion that sequences of mental steps fall outside of patentable subject matter,[IV.20] the CCPA granted protection for computer programs which conducted spectrographic analysis of the concentration of elements in a mixture of gases,[IV.21] mathematically projected three-dimensional figures onto two-dimensional surfaces,[IV.22] and improved seismographic recordings.[IV.23]

In _In re Benson_,[IV.24] the CCPA again reversed the Patent Examiner, granting patent protection for a method of converting from one numerical base to another in conjunction with telephone interconnection processing. The Supreme Court reversed[IV.25] on the ground that the granting of the patent would effectively preempt the algorithm embodied in the method.[IV.26] Although specifically stating that its decision did not preclude patentability for computer programs,[IV.27] the opinion created great uncertainty about the permissible scope of such protection. The uncertainty heightened when the Supreme Court overturned the next case in which the CCPA granted patent protection for a computer protection, this time a program for controlling automatic banking devices.[IV.28] The Supreme Court reversed the granting of a patent not on the ground that the program preempted an algorithm but because the invention was obvious[IV.29]; the Court specifically noted that its _Benson_ holding was limited.[IV.30] The Supreme Court again reversed the CCPA in _Parker v. Flook_,[IV.31] holding that an invention embodying a mathematical algorithm for using an array of variables to update alarm limits could not be protected because the algorithm, like a scientific principle, is assumed to be in the prior art.[IV.32] As in its other computer program cases, the Court noted that it was not holding that computer programs could not be patented.[IV.33]

In the aftermath of these cases, the CCPA continued to grant patents for computer programs,[IV.34] though perhaps with greater reserve.[IV.35] The Patent and Trademark Office saw these cases as conflicting with the Supreme Court's earlier pronouncements.[IV.35] In 1980, the Supreme Court granted certiorari in two such cases. In the first, the CCPA had granted a patent for a computer program embodied in firmware which made it possible to communicate directly with internal registers in the computer.[IV.36] An equally divided Supreme Court affirmed without commenting on the merits. [IV.37] In the other case, Diamond v. Diehr,[IV.38] the CCPA had granted a patent for a rubber curing process which used a programmed computer to determine the precise curing time.[IV.39] In a 5-4 decision, the Supreme Court affirmed. Viewing the claim "as a whole," the Court concluded that the process at issue was no different from other historically patentable industrial processes which transform one article, in this case raw, uncured synthetic rubber, into a different state.[IV.40] The Court did not think the claim was any less patentable because it employed a mathematical algorithm in the process.[IV.40] It distinguished Parker v. Flook [IV.42] on the ground that the invention used an algorithm simply to generate an alarm limit which is merely a numerical value; in effect, a patent would have preempted use of that formula. By contrast, a patent on the process in Diamond v. Diehr would only preempt others from using the complex series of steps in the process.

Since the decision in Diamond v. Diehr, the Patent and Trademark Office has taken a more liberal attitude toward patentability of computer program applications.[IV.43] The CCPA has cautiously expanded patent protection for computer programs. In particular, its decision in In re Pardo[IV.44] portends greater patent protection for computer operating system programs. The inventor in In re Pardo sought patent protection for an invention which converted a computer from a sequential processor to one that is not dependent on the order in which it receives program steps. The CCPA held that the fact that the invention uses an algorithm does not render it invalid.[IV.45] The court concluded that the application properly recited a process for controlling the internal operations of a computer and did not fall within one of the judicially determined exceptions to the patent law.[IV.46]

Despite the availability of patent protection for those computer programs satisfying the requirements of the patent law, there are two principal factors discouraging computer programmers from seeking patent protection. First, in contrast to copyright law which requires only that computer programs be original, the patent law has a relatively high statutory threshold; a computer program must be novel, non-obvious, and useful in order to qualify for protection. Related to this factor, the standards for novelity, non-obviousness, and usefulness with respect to computer programs are still relatively uncertain.

The other major impediment to patenting computer programs is the great cost and long delay associated with obtaining patent protection. A prospective patent owner must first prepare a careful application. The application must then go through a lengthy substantive review by the Patent and Trademark Office.[IV.47] Given the great speed with which computer technology is evolving, the advantages of going through this process are significant for only a small portion of programs developed.

C. Trade Secret Law

Common law and state statutes may be used to protect trade secrets embodied in computer software.[IV.48] Although trade secret law varies from state to state, a widely adopted definition of a trade secret is:

> any formula, pattern, device or compilation of information which is used in one's business, and which gives him an opportunity to obtain an advantage over competitors who do not know or use it.
>
> The subject matter of a trade secret must be secret so that, except by the use of improper means, there would be difficulty in acquiring the information. . . . Some factors to be considered in determining whether given information is one's trade secret are: (1) the extent to which the information is known outside of his business; (2) the extent to which it is known by employees and others involved in his business; (3) the extent of measures taken by him to guard the secrecy of the information; (4) the value of the information to him in developing the information; (5) the amount of effort or money expended by him in developing the information; (6) the ease or difficulty with which the information could be properly acquired or duplicated by others.[IV.49]

Thus, in order to obtain trade secret protection, a business must establish that its information is novel, valuable in the trade or business, and secret. The novelty element requires that the information be more than that commonly known in the trade, though it need not achieve the threshold level of novelty required for patent protection.[IV.50] With regard to computer programs, it is sufficient that the program apply commonly known concepts to a new function[IV.51] or embody a novel combination of generally known concepts.[IV.52] The element of value is usually presumed if the information is used in the business.[IV.53]

With regard to computer programs, particularly those that are mass marketed, the primary obstacle to obtaining trade secret protection is the secrecy requirement.

Although absolute secrecy is not required to obtain trade secret protection,[IV.54] the secrecy element typically requires that a business endeavor to take in-house measures to minimize disclosure and restrict dissemination of the information in products distributed to end-users. Computer software firms typically distribute their products through limited licensing agreements rather than through sales contracts in order to satisfy the secrecy requirement of trade secret law. There is concern, however, that such agreements are unenforceable as contracts of adhesion[IV.55] and that mass distribution of this type is inconsistent with the requirement of secrecy.[IV.56] Commentators suggest that businesses may be able to retain trade secret protection for mass marketed computer programs that are disseminated in object code only.[IV.57]

A business which satisfies the above requirements can prevent those who discover the trade secret by improper means or from a third party from using the trade secret as well as recover damages.[IV.58] A firm is not protected, however, from one who obtains a protected product through permissible means, e.g., unrestricted licensing, and through reverse engineering, discovers the valuable information contained therein.[IV.59]

Furthermore, trade secret protection for computer software may be preempted by the Copyright Act.[IV.60]

Although the House Committee Report indicates that Congress did not intend to preempt state unfair competition and trade secrecy law,[IV.61] the courts have yet to achieve unanimity on this point.[IV.62]

A number of practical considerations also limit the value of trade secret protection. As matters of common law and state statutes, the requirements and scope of trade secret protection vary across states. More significantly, it is difficult for licensors of computer software, particularly for programs which are mass marketed, to monitor and enforce licensing agreements. Moreover, proving violation of trade secret can be especially difficult with regard to computer programs.[IV.63]

D. Semiconductor Chip Protection Act of 1984

In 1984, Congress enacted legislation establishing a new category of legal protection for intellectual work embodied in semiconductor chips (mask works).[IV.64] Since computer software may be fixed in such works (e.g., the fixing of object code in ROMs), the Semiconductor Chip Protection Act (SCPA) provides yet another means by which programmers may protect the intellectual work contained in computer software. The SCPA provides a 10-year term of protection.[IV.65]

Unlike the other forms of protection of computer software, the SCPA is tailored to the unique attributes of the new and evolving technology of semiconductors.[IV.66] For example, the SCPA permits reverse engineering for the purpose of developing improved versions of mask works.[IV.67] It also provides an exception for innocent infringement.[IV.68]

V. Legal Protection for Computer Operating Systems

Having reviewed the technology and economics of computer products, the principal market failures affecting the provision of computer software, and the various forms of legal protection potentially available for computer programs, we are prepared to assess the efficacy of the prevailing legal regime. This Part presents an economic analysis of legal protection for the intellectual work embodied in computer operating systems. Drawing on the economic framework developed in Part III, Section A assesses the need for legal protection. Section B then analyzes the costs and benefits of the primary mode of legal protection for operating systems in the United States -- traditional copyright doctrine. Section C outlines an alternative regime of legal protection for operating systems that better promotes key economic objectives -- innovation and diffusion of improved technologies, standardization of products capable of generating network externalities, and vigorous competition.

A. The Need for Legal Protection

According to the economic framework developed in Part III, legal protection can alleviate the public goods problem inherent in intellectual work. Because legal protection can be costly and have undesirable effects, policymakers should first assess the severity of the public goods problem in particular settings and evaluate offsetting factors.[V.1] Subsection 1 discusses the severity of the appropriability problem in the context of computer operating systems. Subsection 2 then evaluates offsetting factors that reduce the severity of this public goods problem. Subsection 3 seeks to clarify the need for legal protection for operating systems.

1. Appropriability of Investment in Research and Development

a. Development Costs

The time and cost required to develop, maintain, and improve operating systems vary widely depending principally upon the type and size of the computer system for which the operating system is designed.[V.2] Operating systems for the low-end or microcomputer sector of the market have short development cycles and relatively low research and development costs relative to the volume of such units produced.[V.3] Operating systems for the high-end or mainframe sector of the market have long development and operating phases, and substantial research and development costs.[V.4]

b. Copying Costs

The cost of copying operating systems also varies greatly across computer systems. Even though crucial components of some operating systems are coded in ROM, the cost of copying operating systems for microcomputers is low, especially when considered in relation to the size of the market that a copier can enter. In the absence of copyright protection for microcomputer operating systems, many firms would introduce microcomputers using copies of the major operating system components.[V.5]

Because of the greater complexity of mainframe operating systems and the more complex interfaces between mainframe hardware and operating systems, the costs of copying these operating systems is relatively high, though no where near as high as the cost of developing the operating systems. Given the lower volumes in the mainframe market and the need to maintain and update operating system software, it is not clear that copying would be rampant in this sector of the market in the absence of copyright protection.

c. Appropriability of Investment

On high-end and other low volume products, licensing in conjunction with trade secret protection is a viable option for capturing rewards from investments in operating system research and development. Because these systems require maintenance and other services, developers of operating systems implemented on mainframes and minicomputers can assure themselves of a steady flow of revenue through services tied to their products. A recent study of the industry forecasts that even "[t]hose whose operating systems are now in the public domain will use various mechanisms such as renaming, redistributing the functions, rewriting major sections, and implementing microcode to insure that almost all customers will have to pay some license charges for their operating systems."[V.6]

In the mini and microcomputer sectors of the market, the low cost of copying operating systems does not necessarily mean that developers of operating systems cannot appropriate a sizeable portion of the benefits of their development investment. The advantage of being the first to introduce a product featuring a new operating system can enable pioneering firms to establish strong market positions.[V.7] In a static marketplace, this advantage would be quickly competed away by copiers. But given the speed with which computer technology becomes obsolete, first movers can reap a large share of the benefits of an innovation and, through continuing research, take the lead on improvements and new products.

The substantial network externalities flowing from mini and microcomputer operating systems[V.8] provide an added advantage to firms with good reputations and large distribution channels. Such companies can use their brand recognition to benefit from the network externalities even in the absence of legal protection for their products.[V.9] Since they can get trademark protection for their products, they become associated with the industry standard if their product standard is widely adopted. In a market in which consumers value standardization, these factors provide "dominant" firms with a limited ability to charge premium prices and to tie sales of complementary products.[V.10]

2. Offsetting Factors

a. Research Consortia

Investors in research and development can internalize a portion of the free-rider problem by developing products together. Through joint ventures, they can share the costs of developing operating systems.[V.11] Of course, this does not prevent those who do not contribute to the venture from copying the operating systems that are developed. Those who do participate, however, spread the risk of research and development and gain earlier access to the design of the system.[V.12] Moreover, because of the network externalities associated with operating systems, the entire industry benefits from the venture to the extent collaboration produces clear industry standards.

b. Government Subsidies

Government subsidies can substitute for direct commercial profits as a means of promoting research and development. The Federal Government generally funds more than one-third of all research and development in the United States.[V.13] These subsidies have been and continue to be an important source of research support in the computer industry.[V.14] The Department of Defense currently has major projects to develop integrated and automated software design tools,[V.15] multi-processing, and artificial intelligence.[V.16] The National Aeronautic and Space Administration is spending $8 billion over the next decade to develop network operating systems and software tools for the space station project.[V.17] Although these projects are directed toward military and space rather than commercial applications, they will likely have some spillover into the commerical sphere.

c. University Research

Since the results of most academic research is in the public domain, university research contributes to the development of operating systems regardless of the availability of legal protection for operating systems. Universities in the United States and abroad conduct extensive software research and development. A recent study notes that operating system research is one of the principal areas of study at leading U.S., European, and Japanese universities.[V.18]

3. The Presence of Network Externalities

The principal source of network externalities in the operating system field flows from interchangeability of complementary products, most importantly application programs. Since most application software for mainframes is specific to the computer, mainframe operating systems do not feature significant network externalities. To the extent such computers offer network externalities, it is through their interface connections which permit substitutability of peripheral equipment. The operating systems are typically not the critical element in this form of compatibility.

By contrast, mini and microcomputers offer significant network externalities. As these types of computers have proliferated, many firms, both large and small, have developed application programs to run the various operating systems. A survey of computer users indicates that software availability is the most important factor in selecting a make and model of new mini or microcomputer.[V.19]

4. <u>Should Operating Systems be Protectable?</u>

The primary focus of economic analysis in assessing the need for legal protection is to determine whether an adequate level of innovation would be forthcoming in the absence of protection. The foregoing indicates that there is a relatively minor public goods problem in the high end (mainframe sector) of the operating system market and a potentially significant problem in the middle and lower end of the market; but that numerous other factors enable producers to reap reward from inventive efforts and otherwise encourage research and development in computer operating systems.

Moreover, the nature of mini and microcomputer operating systems as both formats for computer systems and creative inventions further suggests that such operating systems would be forthcoming even in the absence of copyright protection. An analogy to the development of railroad transportation helps to illuminate this point. By the mid-nineteenth century, the basic technologies for developing railroad transportation existed. Steam engines capable of moving great weights had been developed. Grades of steel capable of withstanding great force were available. The remaining step to creating a railroad system was to choose a standard gauge of track to which railroad equipment could be conformed. Although choice of an optimal track gauge depended upon certain factors -- e.g., tensile strength of the steel, size of materials to be transported -- the choice among competing gauges was relatively simple in comparison to the development of the steam engine and steel technologies. Given the great promise of a rail system to the developing economies of the nineteenth century, builders of rail equipment probably did not need a right to exclusive use of a particular railroad gauge in order to enter the railroad car market.

Similarly, to a large extent, the technological advances necessary for the development of microcomputers had already taken place by the mid-1970's. The last major step was the choice of an operating system that -- like a railroad gauge in the rail transportation industry -- would serve as a standard for firms wishing to build computer systems. Seen in this light, the desire to create a hardware market would seem incentive enough to encourage development of a standard operating system.

A crucial policy consideration is, therefore, the extent to which operating systems are merely formats (as opposed to creative inventions.) There is no question that some operating systems are more useful than others. Since innovation in operating systems might affect the competitiveness of the U.S. computer industry and because innovation is not costless, caution suggests implementing a minimally disruptive system of legal protection. Moreover, even if no protection was warranted to generate the principal operating systems currently in use, we still might need legal protection to encourage future generations of operating systems. The next section turns to the analysis of copyright protection as a primary means for regulating the market for computer operating systems.

B. The Inappropriateness of Copyright Protection

This Section evaluates the effects of copyright protection for operating systems on the functioning of computer software and complementary hardware markets. Subsection 1 analyzes the effects of copyright protection on innovation and diffusion of new operating systems and complementary hardware products. Subsection 2 discusses the effects of copyright protection on the realization of network externalities flowing from standardized operating systems. Subsection 3 analyzes the effects of copyright protection on competition in the computer industry. Subsection 4 summarizes the effects of copyright protection for operating systems.

Given that the public goods and network externality problems associated with operating systems in the mainframe sector of the market are minimal, there does not seem to be any justification for bestowing copyright protection on these products. Consequently, we focus below on the effects of copyright protection on the mini and microcomputer sectors of the operating system market.

1. Effects on Innovation and Diffusion

As Parts I and II discussed, the computer industry consists of overlapping markets. Part III described how the availability of legal protection at particular levels of a multi-level industry can affect innovation at other levels through diffusion mechanisms. In order to assess the effects of copyright protection for operating systems on innovation, therefore, we must look not only at its effects on the incentives to invest in research and development of operating systems but also its effects on innovation in and diffusion of complementary products.

a. Operating Systems

In order to encourage socially desirable technological innovation, legal protection should be tailored to protect the socially valuable aspect of the intellectual work. Unlike traditional subjects of copyright protection -- literary and artistic works -- computer operating systems are not valued for their expression per se. Operating systems create value through their utilitarian functions -- their ability to direct the inner workings of computer systems.[V.20] Yet copyright law only protects the expression of an idea rather than the idea itself.[V.21] Therefore, copyright does not protect the valuable part of operating systems.[V.22] Consequently, copyright protection does not in general encourage software developers to invent better operating systems.

A hypothetical example illustrates how copyright protection for operating systems might discourage socially valuable innovation while encouraging wasteful research and development expenditures on emulating existing industry standards. Suppose that David Corporation, a relatively small computer company, invents HAL, an extremely valuable microcomputer operating system. David writes some basic application programs for the HAL system and introduces it with a modest marketing campaign. Computer magazines give the HAL system favorable reviews. As the market for HAL grows, application software companies begin to develop software packages to run on the innovative and powerful HAL operating system.

Goliath Corporation, a major computer manufacturer, has carefully watched the favorable reception that David's HAL system has received in the marketplace. Goliath decides to enter the market with its own system and considers its options: (1) negotiating with David for a license to copy the HAL operating system; or (2) developing a similar system on its own. Because copyright protects only the expression and not the ideas of the innovating firm, Goliath can easily develop a product embodying the ideas contained in HAL in a different form of expression without infringing David's copyright. Therefore, it would be bad business judgment for Goliath to pay for something it could get for free (or close to it). Moreover, Goliath could use its reputation and wide distribution channels to establish its system as the industry standard. This has the added benefit to Goliath of squeezing David out of the market.

Goliath's computer programmers and engineers develop the 2001 computer operating system. The 2001 uses the same ideas as the HAL system but programs designed to run on the

2001 will not run on the HAL system. In addition, the programmers develop a large library of application programs to run on the 2001. Goliath also encourages independent software firms to develop programs to run on its version of the HAL system. Goliath releases its 2001 system with great fanfare. Consumers see that the 2001 will become the industry standard and the prophesy is quickly fulfilled.

Unfortunately -- from the perspective of a society seeking to promote the development of improved operating systems -- David does not come back to slay Goliath in this story. Quite the contrary. The combination of inappropriate legal protection and strong network externalities give even a lazy giant more than enough strength to overcome an innovative, though smaller, competitor.

Even if David wanted to switch to the 2001 system, it would be frustrated by Goliath's copyright protection. While copyright did not protect David's ideas, it does protect Goliath's expression. In order to compete, David must invest substantial resources in developing a compatible, though non-infringing, system. Thus, copyright now encourages David to devote its research and development efforts to a wasteful effort to emulate the industry standard.

It might be thought from this hypothetical that even though copyright protection discourages small inventors, it does protect operating systems developed by firms capable of reaping network externalities. Copyright does protect an operating system introduced by a firm capable of starting a bandwagon. But it still does not protect the ideas embodied in an innovative operating system. Therefore, we could tell another story of how a dominant firm that through painstaking research creates a brilliant operating system has its rewards whittled away by clever copiers who build compatible, yet non-infringing, operating systems. If the costs of emulating are low, the dominant firm might not be able to recover its research and development costs. If they are not, then resources are wasted on efforts to emulate an industry standard.

b. Complementary Products

Because of network externalities flowing from computer operating systems, it is difficult for hardware manufacturers to market computer systems unless they can offer one of the *de facto* industry standard operating systems.[V.23] A number of factors limit their ability to gain access to an industry standard that is protected by copyright. As a means of sustaining a dominant position in the marketplace,[V.24] the owner of an operating system copyright might limit its licensing practices as a means of locking consumers into its hardware and other products and discouraging other firms from developing directly competing products.[V.25] In order to sell their hardware products to such customers, competitors must emulate the industry standard without infringing its owner's copyright. This is costly and may fail to achieve a satisfactory level of compatibility;[V.26] moreover, the follower takes the risk that the dominant firm will switch operating systems, thereby leaving followers out in the cold.[V.27]

Even if the owner of the copyright to the industry standard is willing to license it, the transaction costs of doing so may be too high for many smaller companies with good ideas for improving complementary products to sign on.[V.28] In the microcomputer market, there are indications that IBM's dominant position and refusal to license its proprietary software have slowed the industry's move toward lower cost, mass distributed computer systems.[V.29]

A major effect of granting copyright protection to operating systems, therefore, is to decrease the rate of return of hardware manufacturers wishing to compete with a dominant computer manufacturer. This in turn reduces the incentive of these firms to develop and improve hardware technology. Thus, by not permitting hardware manufacturers low cost access to de facto industry standard operating systems, copyright law chokes off innovation and diffusion of computer hardware products.

2. Realization of Network Externalities

Ordinarily, exact duplication is not required to reap the benefits of an idea. Therefore, copyright does not exact too high a cost for such long-lived protection.[V.30] In the case of computer operating systems, however, exact duplication is essential to achieving the ability to run application programs designed for a particular operation system.[V.31] In effect, therefore, copyright protection discourages realization of external benefits flowing from broad diffusion of an industry standard by severely limiting access to that standard.

The members of CONTU were aware of the possibility that granting copyright for computer software might invoke the idea/expression identity exception to copyright protection. "In the computer context, [the] idea/expression identity means that when specific instructions, even though previously copyrighted, are the only and essential means of accomplishing a given task, their later use by another will not amount to an infringement."[V.32] In defining a test for this defense to infringement of operating systems, however, the Third Circuit Court of Appeals erected a virtually insurmountable burden of proof for would-be copiers.[V.33] The court stated that in order for there to be a merger of an idea with its expression, other methods of expressing the idea must be "foreclosed as a practical matter."[V.34] By finding that compatibility with Apple II software and hardware to be "a commercial and competitive objective" of Franklin rather than an essential function of the operating system programs,[V.35] the court failed to appreciate the network externalities inherent in computer operating systems. The competitor can always offer a similar computer product without infringing the dominant firm; but it cannot (at least without substantial, socially worthless effort) offer compatibility without infringing. If potential network externalities are large, then the Third Circuit's decision in Apple has and will continue to extract a high social cost.

The availability of legal protection for operating systems not only prevents firms from offering compatible products but also discourages them from coordinating efforts to establish and develop uniform industry-wide standards. As we discussed in Part III, dominant firms might prefer to offer non-compatible products even when net social welfare would be improved by their adoption of compatible products.[V.36] This observation is consistent with Professor Brock's observations about the evolution of the U.S. computer industry.[V.37] He finds that IBM strongly resisted efforts to standardize computer interfaces and programming languages.[V.38] The availability of copyright protection for computer operating systems strengthens dominant firms' ability to resist standardization by giving them the legal right to enjoin those who adopt their operating systems. It is not surprising, therefore, to find a lack of clearly defined standards to be the major impediment to growth in the U.S. computer industry.[V.39]

3. Conclusions

Our analysis suggests that there is little need for copyright protection for operating systems designed to run on mainframe computers. With respect to mini and micro-computer operating systems, copyright protection makes it more difficult for society to reap the benefits of significant network externalities, discourages and distorts innovation in operating systems and complementary computer hardware, and slows diffusion of computer products. This does not mean that legal protection cannot alleviate the market failures flowing from the public goods and network externality attributes of computer operating systems; rather it means that copyright is ill-suited to remedying these market failures. In the next Section, we draw upon the lessons from the shortcomings of copyright protection to tailor a system of legal protection that can better serve the public interest.

C. Tailoring Legal Protection for Operating Systems

Part of the reason for copyright's inability to promote economic efficiency in the provision of computer products is that the public goods and network externality problems suggest conflicting modes of legal protection. Public goods problems are alleviated by expanding legal protection for intellectual work. On the other hand, external benefits from networks are promoted by facilitating access to a standard. Thus, the difficult policy question is how to promote standardization on the one hand to ensure compatibility while at the same time encouraging continuing innovation (along the entire spectrum from software to hardware) on the other. By closely tailoring legal protection to reward desired innovation while permitting reasonable access to industry standards, it is possible to reach a satisfactory accomodation of these objectives.

In theory, patent law is well suited to protecting the intellectual work contained in computer operating systems.

A patent protects new and useful processes and machines.[V.40] Given the interchangeability of hardware and software,[V.41] it seems logical to protect computer operating systems and dedicated computers embodying a particular operating system with the same form of legal protection. Because patent law protects ideas, those who create patentable operating systems could be better assured of appropriating a substantial portion of the benefits of their efforts.

As we noted in Part IV, it is difficult to obtain patent protection for computer programs. It should be pointed out, however, that although the scope of patent protection for computer software is uncertain, some recent cases have upheld the patentability of computer programs that manipulate the internal operations of a computer.[V.42] Moreover, the importance of network externalities flowing from widespread access to a common mini and microcomputer operating system suggests that legal protection should be hard to come by (and short in duration).

To accomodate the objective of encouraging innovation in operating system technology, Congress should consider creating a hybrid form of patent protection specifically tailored to accomodate the market failures endemic to the provision of computer operating systems.[V.43] As with traditional patent law, the standard for protection should be novelty, non-obviousness, and usefulness; dominant firms (or anyone else) should not be able to "lock-up" an industry standard simply by expressing it in a unique way.

To be feasible, the modified form of patent protection for computer operating systems should be determined on the basis of a timely examination of patent applications. And given the rapid pace of technological change in the computer field and the interest in promoting access to industry standards, patent protection for operating systems should be shorter in duration than traditional patent protection.[V.44]

In order to promote continued innovation in widely-used operating systems, the operating system patent code should, like the Semiconductor Chip Protection Act, permit some limited form of reverse engineering. And like traditional patent law, it should not infringe the hybrid code for consumers to buy a ROM or other device containing a patented operating system and modify it for sale to a third person.[V.45]

In order to facilitate realization of network externalities, the hybrid patent code should contain a flexible compulsory licensing provision.[V.46] Such a provision would promote access to an industry standard while assuring reward to the creator of an innovative and socially valuable operating system. It would also limit the ability of dominant firms in the industry to engage in anticompetitive practices.

The need for compulsory licensing as a means for promoting competition and rewarding innovation is brought into focus by the recent decision of the Ninth Circuit Court of Appeals in Digidyne Corp. v. Data General Corp.[V.47] In Data General, the defendant (Data General), a manufacturer of computers, refused to license its RDOS operating system to firms using a central processing unit other than Data General's "NOVA" system. Recognizing the anticompetitive effects of this practice in a market featuring network externalities,[V.48] the Ninth Circuit held that Data General's licensing practices were an unlawful tying arrangement that violated federal antitrust law.

In light of the strong network externalities flowing from compatibility, computer operating systems serve as "essential facilities" in computer hardware markets.[V.49] Unless a firm can get onto the network, its products will be at a great disadvantage relative to those that can run the vast stock of application programs designed for the industry standard. The royalty rate should be set so as to compensate true innovators for the cost of building a useful "highway" for the market plus a fair profit (adjusted for the risk of failure).[V.50] For high volume products, these rates would probably be low. In the microcomputer market, for example, the rate would probably be less than $1 for access to the major operating systems (assuming that the Apple and IBM operating systems merited hybrid patent protection at all).[V.51]

A patent code for operating systems based on the above outline a preferable balance of the conflicting policy concerns raised by computer operating systems. By providing solid protection for truely innovative and useful operating systems, the code rewards innovation in operating systems. The limits on this regime of protection -- moderate duration, reverse engineering, adaptation -- and the provision for compulsory licensing promote access to operating systems that emerge as industry standards, wide diffusion of computer products, and innovation in hardware products. The code also avoids wasteful expenditure of resources on efforts to emulate an industry standard.

The proposed operating system code would most likely entail somewhat higher administrative costs than the current system. Patent examinations, though streamlined, would be significantly more expensive than the cost of copyright registration. Moreover, compulsory licensing proceedings as well as the cost of monitoring use of protected operating systems would add to the expense of the system. If the royalty rates are small (as the microcomputer example indicates), however, members of the industry could be expected to cooperate in ensuring that patent owners are properly compensated.

VI. Legal Protection for Application Programs

This Part presents an economic analysis of legal protection for intellectual work embodied in application programs. Section A assesses the need for legal protection, and concludes that the ease with which valuable programs can be copied justifies limited legal protection. Section B analyzes the costs and benefits of using traditional copyright doctrine to protect these programs. Section C outlines a *sui generis* form of legal protection that better promotes technological innovation and diffusion of software products.

A. The Need for Legal Protection

At the outset, it will be useful to distinguish between software programs written specifically for particular consumers (contract programs) and general purpose programs that can be used by more than one consumer (software packages). Contract programs do not invoke the public goods problem because the full benefit of the creator's work is realized by the single or few consumers for whom the programs are designed; therefore, the creator can easily limit the distribution of its work and can charge for its full development and maintenance costs.[VI.1] Contract programming has been the mainstay of the software industry during most of its history. With advances in general purpose computing in the past decade, however, software packages have become an important software product, particularly at the low-end or microcomputer sector of the market. Because the contract program sector of the market does not evidence significant public goods problems, we focus upon the software packages sector of the industry below.

The characteristics of software packages vary widely depending principally upon the type of computer system on which the application programs will be run. Software packages for the low-end of the market typically have high unit volumes, short product life cycles, and low or zero maintenance costs. At the middle and upper levels of the equipment size spectrum, software packages are characterized by lower unit volumes, longer product life cycles, and higher maintenance costs.[VI.2]

In the absence of devices to inhibit copying,[VI.3] the cost of copying software packages for most systems is low. Programs are typically stored on disks or tapes that can usually be copied directly on the machine for which they were designed. Even where direct copying is not possible, resourceful computer programmers and engineers can often reverse engineer the programs.

Despite the ease of copying, there are ways by which creators of application program can appropriate their investments in research and development. Creators of low-volume, high-cost software packages for high-end computers can often realize much of their cost on the first units released. The price of these programs can exceed $60,000. Moreover, most of this software is not sold outright but is licensed or leased on a monthly or yearly basis, with separate fees for maintenance, updating, and improvements.[VI.4] The low volumes of these sales make it possible for the creator to obtain trade secret protection for their intellectual work.

Appropriating the cost of developing software packages for microcomputers is more difficult. Prices for these programs -- ranging from $20 to $1000 -- do not allow creators to recover their investment costs on the first units sold. They make their return on high volume sales.[VI.5] Because high volume sales requires them to market these programs widely, however, creators lose the ability to obtain trade secret protection.

Creators of high-volume application packages have sought other means of appropriating their development costs. Some tie maintenance and program enhancement services to their programs.[VI.6] Others use copy-protect devices that prevent or impede copying.[VI.7] However, no failsafe device without some drawbacks has yet been developed. Moreover, these forms of protection have the undesirable effect of limiting the consumer's ability to modify programs.

Thus, the ease with which software packages can be copied suggests that some form of legal protection is needed to assure creators of software packages an adequate return to their investment. It should be emphasized that this concern is most acute in the low-end/microcomputer sector of the market. Creators of low-volume packages for medium-sized and large computer systems can use licensing agreements to protect their intellectual work. Moreover, they can tie maintenance support and enhancement services to their programs in order to reap a stream of payments over the life of the programs. Creators of software packages sold in retail stores do not have these means of protecting their intellectual work from copiers and therefore are in the most need of legal protection.

B. Overinclusiveness of Full Copyright Protection

This Section assesses the efficacy of the current system for protecting intellectual work embodied in application programs. Subsection 1 analyzes the effects of traditional copyright protection on the incentives to innovate and diffuse computer technologies. Subsection 2 assesses the role of network externalities in the application software market. Subsection 3 discusses the effects of traditional copyright protection on competition in markets for computer hardware and software.

1. <u>Effects on Innovation and Diffusion</u>

In a recent survey of two thousand diverse organizations, <u>Software News</u> found that "documentation" and "ease of use" are the most important selection criteria among application software users.[VI.8] These results suggest that expression is a critical factor in the value of application software. Thus, copyright protection does protect the socially valuable aspect of the intellectual work embodied in application programs.

While copyright does not effectively prevent copiers from making single or a few copies (because of detection and enforcement costs), it does inhibit competing firms from reproducing works on a mass scale.[VI.9] Therefore, copyright greatly enhances the ability of creators of high-volume application programs to capture a substantial portion of the social value of their work.

Copyright's long duration of coverage and limitations on adaptation of protected works, however, inhibit the creation of improvements on existing programs, thereby constraining diffusion of such programs. One commentator notes that "[n]ew computer programs . . . often rely on existing programs, and this reliance will surely increase when programmers reach the stage of creating new programs by computer instead of human intellectual effort."[VI.10] In addition, innovation in application packages can take the form of combining existing application programs in a useful way.[VI.11] Traditional copyright doctrine does not easily accomodate this form of innovation. Except where the programs to be enhanced, combined, or synthesized are in the public domain or the proprietary library of the would-be creator, she must license the desired programs. The transaction costs associated with licensing in this context can be prohibitive.

2. Realization of Network Externalities

Application programs do create network externalities; but of a different type than operating systems. As users become familiar with a particular type of application program -- for example, a word processing program or a spreadsheet -- the value of their knowledge is greater, the greater is the adoption of that product. This network externality is similar to that created by a standardized typewriter keyboard.[VI.12] In effect, wide adoption of common application programs means that users' acquired skills are transferable to a different workplace.[VI.13] But unlike the QWERTY example, the switching costs associated with application programs are relatively small because of the availability of many application programs in the major formats.

The availability of copyright protection for dominant application packages will discourage realization of some of the benefits of these network externalities by enhancing the ability of dominant firms to act unilaterally in establishing proprietary standard programs. For example, dominant firms might try to have their word processing packages, spreadsheets, or programming language adopted, even though another package of comparable capability has achieved popularity on the market. Such firms may also undermine efforts to establish uniform voluntary standards.[VI.14]

3. <u>Effects on Competition</u>

Unlike the case of operating systems, the potential for consumer lock-in and tying with application programs is not great under a copyright regime. Although it is theoretically possible for the creator of an application package to tie its sale to hardware and operating systems, this market power is greatly limited by the availability of close substitutes. Unlike with operating systems, exact coding is not essential to producing a similar competing application program.

4. Conclusions

Copyright protection does promote innovation in application programs, though not without some inhibiting effects on diffusion and the realization of network externalities. The next Section suggests ways that legal protection might be better tailored so as to retain the beneficial effects of copyright protection while reducing some of its negative impacts.

C. Tailoring Legal Protection for Application Programs

As with legal protection for operating systems, Congress should consider creating a special form of legal protection for application programs.[VI.15] Given the importance of improving existing programs as a primary mode of technological innovation and the presence of some network externalities, legal protection should be significantly shorter in duration than that under traditional copyright law. The relatively short commercial life of most application programs[VI.16] indicates that legal protection should be correspondingly short.[VI.17]

The regime for protecting application programs should also allow for reverse engineering. In designing legal protection for semiconductor chips, Congress recognized the importance of reverse engineering in enabling researchers to advance a field in which innovations are cummulative.[VI.18] A limited reverse engineering provision in the application software code would similarly promote the advancement of application software technology.[VI.19]

Congress should also consider the desirability of a limited form of compulsory licensing of application packages. In order to realize the benefits of network externalities and to promote creativity in the integration of software programs, it would seem worthwhile to allow limited access to application programs, particularly those that emerge as industry standards. This could be achieved without dulling primary creative incentives by delaying the availability of compulsory licensing for a limited period to allow the creator of the program to reap the rewards of commercial success.

VII. Concluding Remarks

This Article has highlighted two serious flaws in the analysis which guided Congress in adopting copyright as the primary means for protecting intellectual work embodied in computer software. With regard to the public goods problem associated with technological innovation, CONTU failed to distinguish among software products and to assess carefully the need for additional legal protection. Second, CONTU overlooked the fact that operating systems serve as product standards which are capable of producing substantial network externalities.

When these considerations are taken into account, a very different set of policy recommendations emerge. Legal protection for mini and microcomputer operating systems must reward important innovations without bestowing pure monopolies to their innovators. It was shown that copyright protection of operating systems exacerbates the market failures endemic to the market for operating systems; and that a hybrid form of patent protection with the availability of compulsory licensing had the potential to satisfy these constraints. Legal protection for mass marketed application programs, on the other hand, is amenable to copyright protection; but of much shorter duration and featuring more flexibility than traditional copyright doctrine.

The analysis and recommendations of this Article set out an agenda for further research. We have highlighted the principal market failures affecting the provision of computer software and sketched out their policy implications. But there is much to be learned by studying each of the factors discussed in this paper in greater detail and with the aid of technological expertise. What is needed is for Congress to appoint a new commission comprised of computer scientists, economists, and lawyers to study these problems in depth. The commission should assess a broad range of policy options -- from hybid forms of legal protection to direct ways of coordinating industry development -- because the problems identified here are not exclusively or necessarily best addressed solely through traditional modes of legal protection.

Swiftness in taking action is all the more important because it seems that the United States is quickly leading the world community toward an ill-advised system of legal protection for computer software.[VII.1] Traditional copyright protection has yet to become entrenched as the world standard, but the possibility is not far off.[VII.2] The United States can play a important role in directing the way toward a sounder course, but only through careful analysis of the underlying problems.

FOOTNOTES

1. See National Commission on New Technological Uses of Copyrighted Works, Final Report 3 (1979) (hereinafter cited as CONTU Final Report); Note, Copyrighting Object Code: Applying Old Legal Tools to New Technologies, 2 Computer/Law J. 421 (1983).

2. Congress also requested CONTU to study the intellectual property issues raised by photocopying and computer data bases.

3. See CONTU Final Report, supra note 1 at 1.

4. See infra Part IV(A).

5. Congress conducted hearings in April, 1986 to consider the implications of these changes for public policy. To accompany these hearings, the Office of Technology Assessment prepared a study entitled Intellectual Property Rights in an Age of Electronics and Information.

6. Computer and Business Equipment Manufacturing Association, Computer and Business Equipment Marketing and Forecast Data Book 87 (1985) (hereinafter cited as CBEMA Data Book).

7. Id.

8. Id. at 85.

9. See Comment, Copyright Protection for Programs Stored in Computer Chips: Competing with IBM and Apple, 7 Hamline L. Rev. 103, 122 (noting that "copyright protection for operating systems stored in [read-only memory] . . . makes it difficult, if not impossible, for small manufacturers to compete with IBM and Apple").

10. See B. Kelly & D. Grimes, IBM PC Compatible Computer Directory xx, xxi (1985) (explaining that no system can be totally compatible with the IBM PC without infringing copyright laws and noting that some compatible computer manufacturers "invest a great deal of time amd effort to ensure that their systems will support the major software on the market [that is formatted for the IBM PC]); Miller & McMillan, Special Report: IBM Compatibility, Popular Computing 104, 108-09 (April 1984).

11. See Wilson, Computers: When Will the Slump End? Business Week 58, 58 (April 21, 1986) (noting that "[t]he biggest impediment to growth [in the computer industry] . . . is the lack of a clearly defined set of hardware and software

standards to build [computer] networks); Pournelle, _The Micro Revolution: Clearing a Path_, Popular Computing 81, 81 (July 1984) (lamenting that the "16-bit operating system jungle offers confusion, not standardization").

12. _See_ CONTU Final Report, _supra_ note 1 at 9-12.

13. CONTU did recognize that that the computer industry lacked clear product standards. _See_ CONTU Final Report, _supra_ note 1, at 13. They did not, however, comprehend the full import of the standardization issue nor did they realize its connection to the design of rules protecting intellectual property. The blame for this critical oversight should not be heaped solely on the CONTU members. Their panel of expert economists did not address the standardization/compatibility issue in its report to the Commission. _See_ Y. Braunstein, D. Fischer, J. Ordover, & W. Baumol, _Economics of Property Rights as Applied to Computer Software and Data Bases_, (National Technical Information Service, U.S. Department of Commerce, PB-268 787, June 1977) (Report prepared for the National Commission on New Technological Uses) (hereinafter cited as "Economics of Property Rights").

14. The use of an economic framework for analyzing federal protection for intellectual property is generally consistent

with the intellectual property clause of the U.S. Constitution. The stated constitutional purpose of Congress' power to grant legal protection to intellectual property is "t[o] promote the Progress of Science and useful Arts, be securing for limited times to Authors . . . the exclusive Right to their . . . Writings." U.S. Const. art. I, Section 8. The Supreme Court has interpreted this language to mean that the benefit accorded the author is a "secondary consideration," United States v. Paramount Pictures, Inc., 334 U.S. 131, 158 (1948), and that the "economic philosophy behind this clause empowering Congress to grant patents and copyrights is the conviction that encouragement of individual effort by personal gain is the best way to advance the public welfare." Mazer v.Stein, 347 U.S. 201, 219 (1954). See Note, Computer Programs and Proposed Revisions of the Patent and Copyright Laws, 81 Harv. L. Rev. 1541, 1549 (1968); Breyer, The Uneasy Case for Copyright: A Study of Copyright in Books, Photocopies, and Computer Programs, 84 Harv. L. Rev. 281, 284-93 (1970); see also Twentieth Century Music Corp. v. Aiken, 422 U.S. 151 (1975) (although the immediate effect of copyright law is to secure a fair return for an author's creative labor, such private motivation "must ultimately serve" to "promot[e] broad public availability" of the fruits of that labor); Kewanee Oil Co. v. Bicron Corp., 416 U.S. 470, 480-81 (1974); United States v. Bily, 406 F.Supp. 726, 730 (E.D.

Pa. 1975) (noting that the copyright system is meant to be "no more extensive than is necessary in the long run to elicit a socially optimal amount of creative activity").

Footnotes for Part I

I.1. See generally E. Feigenbaum & B. McCordruck, The Fifth Generation (1983) (tracing the evolution of computer technology).

I.2. A third important type of software is called microcode. Microcode, the most basic level of software, is simply a set of encoded instructions. Microcode typically substitutes for hardware circuitry that executes the primitive functions of the computer. Because of its close kinship to hardware, microcode is also referred to as firmware.

I.3. See generally Cook, Special Report: Operating System, Popular Computing 111 (Aug. 1984) (reviewing the major operating systems).

Footnotes for Part II

II.1. See Computer Technology Shifts Emphasis to Software: A Special Report 143 Electronics (May 8, 1980).

II.2. See E. Rogers, Diffusion of Home Computers (1982).

II.3. See CBEMA Data Book, supra note 6 at 87.

II.4. Id.

II.5. See Harbridge House, Legal Protection of Computer Software: An Industrial Survey (Report prepared for the National Commission on New Technological Uses of Copyrighted Works, Nov. 1977).

II.6. See U.S. Department of Commerce, International Trade Administration, A Competitive Assessment of the U.S. Software Industry (1984) (hereinafter cited as Software Trade Study).

II.7. See CBEMA Data Book, supra note 6 at 85.

II.8. See Wilson, supra note 11.

Footnotes for Part III

III.1. Market imperfections arise in many forms. See S. Breyer, Regulation and its Reform 15-32 (1982). The more important types are the presence of monopoly or monopsony power, incomplete or incorrect information on the part of economic agents, and externalities. Production externalities are costs or benefits accruing to the producer of a good that are not borne by the seller or the buyer of the good. Industrial pollution (not charged to the manufacturer or otherwise "internalized" by the regulatory/legal system) illustrates a classic externality problem. Goods for which an individual's consumption of the good does not preclude others (who do not pay for the good) from consuming part or all of the good generate consumption externalities. We discuss these types of externalities further below. See infra text accompanying notes III.16-III.17.

III.2. See P. Samuelson & W. Nordhaus, Economics 678 (12th ed. 1985). The focus on efficiency concerns in this article should not be construed as suggesting that distributional issues are not seen as important. Rather it reflects the view that such considerations are better addressed by taxation policy, welfare policies, and other more direct and better targeted means of income redistribution.

III.3. <u>See</u> D. Hemenway, Industrywide Voluntary Product Standards (1975); Katz & Shapiro, <u>Network Externalities, Competition, and Compatibility</u>, 75 Am. Econ. Rev. 424, 424 (1985).

III.4. <u>See</u> Arrow, <u>Economic Welfare and the Allocation of Resources for Invention</u>, in The Rate and Direction of Inventive Activity: Economic and Social Factors (1962).

III.5. <u>See</u> Samuelson, <u>The Pure Theory of Public Expenditure</u>, 36 <u>Rev. Econ. & Stat.</u> 387 (1954).

III.6. <u>See</u> CONTU Final Report, <u>supra</u> note 1, at 9-12.

III.7. <u>See</u> Kaplow, <u>The Patent-Antitrust Intersection: A Reappraisal</u>, 97 Harv. L. Rev. 1813, 1823 - 24 (1984).

III.8. <u>See</u> P. David, <u>New Technology Diffusion, Public Policy, and Industrial Competitiveness</u> 6 (Center for Economic Policy Research, Publication No. 46, April 1985).

III.9. Rosenberg, <u>Factors Affecting the Diffusion of Technology</u>, 10 Explorations in Econ. Hist. 3 (1972).

III.10. <u>See e.g.</u>, Mak & Walton, <u>Steamboats and the Great Productivity Surge in River Transportation</u>, 32 J. Econ.

Hist. 625 (1972); Enos, A Measure of the Rate of Technological Progress in the Petroleum Refining Industry, 6 J. Ind. Econ. 189 (1958).

III.12. See P. David, supra note III.8 at 5 (expressing the concern that increased protection for intellectual property might hinder diffusion).

III.13. Thus, though recognizing the public goods problem inherent in markets for goods embodying intellectual property, some commentators have questioned the desirability of government intervention to correct this market imperfection. See e.g., Breyer, supra note 14; Plant, The Economic Aspects of Copyright in Books, 1 Economica 167 (new series 1934).

III.14. See Schmalensee, Product Differentiation Advantages of Pioneering Brands, 72 Am. Econ. Rev. 349 (1982); Bond & Lean, Consumer Preference, Advertising, and Sales: On the Advantage from Early Entry (Bureau of Economics, U.S. Federal Trade Commission, Working Paper 14, Oct. 1979); J. Bain, Barriers to New Competition (1956).

III.15. See M. Peck, Government Research and Development Subsidies in the American Economy? (Economic Research Institute, Economic Planning Agency, Tokyo, Japan, Discussion paper No. 35, April 1985); Levin, The Semiconductor Industry in Government and Technical Progress: A Cross Industry Study (R. Nelson (ed.) 1982).

III.16. See Katz & Shapiro, supra note III.3.

III.17. See David, Clio and the Economics of QWERTY, 75 Am. Econ. Assoc. Papers and Proceedings 332 (1985).

III.18. See D. Hemenway, Industrywide Voluntary Product Standards (1975).

III.19. See Brock, Competition, Standards and Self-Regulation in the Computer Industry, in Regulating the Product: Quality and Variety 75, 75 (R. Caves & M. Roberts eds. 1974) (noting that "[e]ffective standards greatly facilitate the interchange of data and programs among the machines of different manufacturers and allow the user to combine equipment from several suppliers").

III.20. See Katz & Shapiro, supra note III.3 at 435.

III.21. See id.; see also Brock, supra note III.19 at 78 (noting that "[f]rom the manufacturer's viewpoint, the value of standards depends upon his competitive position. If he is satisfied with his current market share, he will want to differentiate his product as much as possible from competing products.").

III.22. _See_ Katz & Shapiro, _supra_ note III.3 at 435-36.

III.23. _See_ Farrell & Saloner, _Standardization, Compatibility, and Innovation_, 16 Rand J. of Econ. 70 (1985).

III.24. _See_ Farrell & Saloner, _Installed Base and Compatibility: Innovation, Product Preannouncements, and Predation_, 76 Am. Econ. Rev. 940, 940 (1986); Farrell & Saloner, _supra_ note III.23 at 71-72, 75-79 (1985); D. Hemenway, _supra_ note III.18 at 30, 39.

III.25. _See_ Farrell & Saloner, _supra_ note III.23 at 75-79; Arthur, _Competing Technologies and Lock-In by Historical Small Events: The Dynamics of Allocation Under Increasing Returns_ 19-21 (Center for Economic Policy Research, Stanford University, Publication No. 43 (1985).

III.26. _See_ David, _supra_ note III.17.

III.27. _See_ Lecture by Ruth Davis of the Center for Computer Sciences and Technology, National Bureau of Standards, at Harvard University (April 1972), cited in D. Hemenway, _supra_ note III.18 at 39.

III.28. See generally 3 P. Areeda & D. Turner, Antitrust Law ¶ 733 (1978).

III.30. See Kaplow, Extension of Monopoly Power through Leverage, 85 Colum. L. Rev. 515, 517-19 (1985); Posner, The Chicago School of Antitrust Analysis, 127 U. Pa. L. Rev. 925, 926 (1979).

III.31. See Kaplow, supra note III.30 at 530-36.

III.32. See Carlton & Klamer, The Need for Coordination among Firms, with Special Reference to Network Industries, 50 U. Chi. L. Rev. 446 (1983).

III.33. See Ordover & Willig, Antitrust for High-Technology Industries: Assessing Research Joint Ventures and Mergers, 28 J. Law & Econ. 311 (1985); Carlton & Klamer, supra note III.32.

III.34. See Brock, supra note III.19 at 91-94.

III.35. Cf. F.M. Scherer, Industrial Market Structure and Economic Performance 454 (2d ed. 1980) (noting that "[a]n ideal patent system would hand-tailor the life of each patent to the peculiar circumstances of the invention it covers, but this is administratively infeasible").

III.36. See Karjala, Lessons from the Computer Software Debate in Japan, 1984 Ariz. State L.J. 53, 63 (noting interest in Japan for distinguishing interface software from other software in the design of legal protection); But see Davidson, Protecting Computer Software: A Comprehensive Analysis, 1983 Ariz. State L.J. 611, 674 (questioning the advisability of this distinction).

III.37. See, Samuelson, CONTU Revisited: The Case Against Copyright Protection for Computer Programs in Machine-Readable Form, 1984 Duke L.J. 663, 672-89 (highlighting the substitutibility of hardware and software).

III.38. See Cook, Operating Systems in Transition, High Technology 65, 69 (June 1984) (noting that technological advances have led to new products that blur the distinction between operating systems and application programs).

III.39. In this way, the distinction sought here is analogous to the definition of an "essential facility" under antitrust doctrine. See infra note V.48.

Footnotes for Part IV

IV.1 Act of October 19, 1976, Pub. L. No. 94-553, 90 Stat. 2541 (*codified* at 17 U.S.C. sections 101 *et seq*.).

IV.2. 17 U.S.C. § 102(a).

IV.3. 17 U.S.C. § 302(a).

IV.4. 17 U.S.C. § 101.

IV.5. *Id*.

IV.6. *See* H.R. Rep. No. 1476, 94th Cong., 2d Sess. 54, *reprinted in* 1976 U.S. Code Cong. & Ad. News 5659, 5667 ("'literary works' ... includes ... computer programs").

IV.7. 17 U.S.C. § 101.

IV.8. 17 U.S.C. § 117.

IV.9. *Apple Computer, Inc., v. Franklin Computer Corp.*, 714 F.2d 1240 (3d Cir. 1983).

IV.10. *Id*. at 1247-48.

IV.11. 35 U.S.C. § 101.

IV.12. 35 U.S.C. § 112 (1982).

IV.13. See cite ______________. But cf. Sears Robuck & Co. v. Stiffel Company, 376 U.S. 225, 230 (noting that "[o]nce [a] patent issues, it is strictly construed [and] cannot be used to secure any monopoly beyond that contained in the patent" (citations omitted)).

IV.14. In re Yuan, 188 F.2d 377, 380 (C.C.P.A. 1951)

IV.15. O'Reilly v. Morse, 56 U.S. (15 How.) 402 (1854).

IV.16. Eibel Process Co. v. Minnesota & Ontario Paper Co., 261 U.S. 45 (1923).

IV.17. Mackay Co. v. Radio Corp., 306 U.S. 86, 94 (   ).

IV.18. See Moskowitz, The Metamorphosis of Software-Related Invention Patentability, 3 Computer/Law J. 273, 281-82 (1982).

IV.19. Until October 1, 1982, the Court of Customs and Patent Appeals had jurisdiction over decisions of the Board of Patent Appeals. 28 U.S.C. § 1542 (1978), repealed by

Pub.L. 97-164, 96 Stat. 41 (1982). The CCPA was replaced by the Court of Appeals for the Federal Circuit which has exclusive jurisdiction over all patent cases. 28 U.S.C. § 1295(a) (1982).

IV.20. In re Prater, 415 F.2d 1378, 1387 (C.C.P.A.), aff'd on this point on rehearing, 415 F.2d 1393, 1401 (1969).

IV.21. Id. at 1405-06.

IV.22. In re Berhard, 417 F.2d 1395 (C.C.P.A. 1969).

IV.23. In re Musgrave, 431 F.2d 882 (C.C.P.A. 1970).

IV.24. In re Benson, 441 F.2d 682 (C.C.P.A. 1971).

IV.25. Gottschalk v. Benson, 409 U.S. 63 (1972).

IV.26. Id. at 71-72.

IV.27. Id. at 71.

IV.28. Dann v. Johnston, 425 U.S. 219 (1975).

IV.29. Id. at 230.

IV.30. _Id_. at 224.

IV.31. _Parker v. Flook_, 437 U.S. 584 (1978).

IV.32. _Id_. at 595.

IV.33. _Id_.

IV.34. _See_ _In re Johnson_, 589 F.2d 1070 (C.C.P.A. 1978) (granting patent protection for a computer-implemented method of filtering noise from data obtained in seismic measurement).

IV.35. _See_ _e.g._, _In re Sarkar_, 588 F.2d 1330 (C.C.P.A. 1978) (rejecting a claim for a technique for mathematically modeling flow parameters of a river); _In re Gelnovatch_, 595 F.2d 32 (C.C.P.A. 1979) (rejecting a claim for a process for determining values for use in a model of a microwave circuit).

IV.36. _In re Bradley_, 600 F.2d 807 (C.C.P.A. 1979).

IV.37. _Diamond v. Bradley_, 450 U.S. 381 (1981).

IV.38. 450 U.S. 175 (1981).

IV.39. In re Diehr, 602 F.2d 892 (C.C.P.A. 1979).

IV.40. 450 U.S. at 184.

IV.41. Id. at 185-88.

IV.42. 437 U.S. 584 (1978).

IV.43. See Moskowitz, supra note IV.18 at 309-10.

IV.44. 684 F.2d 912 (C.C.P.A. 1982).

IV.45. Id. at 916-17.

IV.46. Id. at 916.

IV.47. See 36 U.S.C. § 131 (1982).

IV.48. See e.g., Uniform Trade Secrets Act.

IV.49. Restatement of Torts § 757, comment b (1939). The reporters of the Second Restatement decided that trade secret law more properly belongs in a separate field of law relating to unfair competition and trade regulation. Consequently, they omitted this section from the Second Restatement of Torts. See 4 Restatement (Second) of Torts, at 1-2 (1977). The definition in text continues to be widely cited. See Restatement (Second) of Torts § 757 app. (1977 & Supp. 1985-86) (collecting cases).

IV.50. See R. Milgrim, Trade Secrets,) 2.08 (1978).

IV.51. See Structural Dynamics Research Corp. v. Engineering Mechanics Research Corp., 401 F.Supp 1102 (E.D.Mich. 1975).

IV.52. See Imperial Chemical Indus. Ltd. v. National Distillers & Chem. Corp., 342 F.2d 737, 742 (2d Cir. 1965); Com-Share v. Computer Complex, Inc. 338 F.Supp 1229, 1234 (E.D.Mich. 1971).

IV.53. Gilburne & Johnston, Trade Secret Protection for Software Generally and in the Mass Market, 3 Computer/Law J. 211, 215 (1982).

IV.54. Q-Co Industries v. Hoffman, 625 F.Supp 608, 617 (S.D.N.Y. 1985); Data General Corp. v. Digital Computer Controls, Inc., 297 A.2d 437, 439 (Del. 1972).

IV.55. See Gilburne & Johnston, supra note IV.53 at 229.

IV.56. See id. at 229-37.

IV.57. See id.; McGrody, Protection of Computer Software -- An Update and Practical Synthesis, 20 Hous. L. Rev. 1033, 1063 (19).

IV.58. Restatement of Torts §§ 757, 758 (1939).

IV.59. See Gilburne & Johnston, supra note IV.53 at 235-37. Of course, the holder of the trade secret could prevent reverse engineering by requiring, through contract, that users not disassemble or disseminate the program. This, however, adds to the cost and complexity of licensing computer software.

IV.60. See 17 U.S.C. § 301 (1982). The Supreme Court has ruled that trade secret law is not preempted by patent law. Kewannee Oil Co. v. Bicron Corp., 416 U.S. 470 (1974).

IV.61. See House Report No. 96-1307, Part I, Committee on the Judiciary, 96th Cong., 2d Sess., 23-24 (1980).

IV.62. Compare Brignoli v. Balch Hardy and Scheinman, Inc., 645 F.Supp. 1201, 1205 (S.D.N.Y. 1986) (holding no preemption) with Videotronics, Inc. v. Bend Electronics, 564 F.Supp. 1471, 1477 (D.Nev. 1983) (finding trade secret law preempted). Cf. Warrington Assoc., Inc. v. Real-Time Engineering Systems, Inc., 522 F.Supp. 367, 369 (N.D. Ill.

1981) (recognizing that trade secret law is not preempted by the Copyright Act but noting that registration of works pursuant to copyright law might destroy the secrecy required for trade secret protection).

IV.63. See Gilburne & Johnston, supra note IV.53 at 255-63.

IV.64. See Semiconductor Chip Protection Act of 1984, 17 U.S.C. §§ 901-14 (Supp. II 1984). Section 901(a)(2) of the Act defines a "mask work" as:

> a series of related images, however fixed or encoded --
>
> (A) having or representing the predetermined, three-dimensional pattern of metallic, insulating, or semiconductor material present or removed from the layers of a semiconductor chip product; and
> (B) in which series the relation of the images to one another is that each image has the pattern of the surface of one form of the semiconductor chip product . . . .

17 U.S.C. § 902 (a)(2) (Supp. II 1984). "Semi-conductor chip product" is defined as:

> the final or intermediate form of any product--
>
> (A) having two or more layers of metallic, insulating, or semiconductor material, deposited or otherwise placed on, or etched away or otherwise removed from, a piece of semiconductor material in accordance with a predetermined pattern; and
> (B) intended to perform electronic circuitry functions . . .

17 U.S.C. § 901(a)(1).

IV.65. 17 U.S.C. § 904 (Supp. II 1984).

IV.66. _See generally_ R. Stern, Semiconductor Chip Protection (1986).

IV.67. SCPA, 17 U.S.C. § 908(a).

IV.68. SCPA, 17 U.S.C. § 907.

Footnotes for Part V

V.1. See generally Breyer, supra note 14 (scrutinizing the need for copyright protection for books, photocopies, and computer programs).

V.2. See generally, Software Trade Study, supra note II.6 (describing the economics of software development).

V.3. Apple Corporation spent approximately $740,000 to develop the 14 main operating system and application programs use in its Apple-II computer system. See Apple Computer v. Franklin Computer, 714 F.2d 1240, 1243 (3rd Cir. 1983). By November 1984, Apple had sold more than 2,000,000 Apple II systems. See Rubin, Special Report: The Life & Death & Life of the Apple II, Personal Computing 72, 72 (Feb. 1985). Apple's operating system development cost per unit sold on the Apple II, therefore, is substantially less than $1.

IBM paid less than this to develop the operating system it uses in its PC line. The main portion of its operating system was licensed for $50,000. See Rubin & Stehlo, Special Report: Why So Many Computer Look Like the "IBM Standard", Personal Computing 52, 54 (Mar. 1984). IBM introduced its proprietary link in the system by coding its Basic Input/Output System on a ROM. See B. Kelly & D. Grimes, IBM PC Compatible Computer Directory xx (1985).

V.4. <u>See</u> Software Trade Study, <u>supra</u> note II.6.

V.5. <u>See</u> Rubin & Strehlo, <u>supra</u> note V.3.

V.6. The Future of Information Processing Technology 75 (S. Andriole (ed.) 1985).

V.7. <u>See</u> <u>supra</u>, text accompanying note III.14.

V.8. Since independent software producers target their application programs for operating systems that are likely to achieve and maintain a substantial market share, computer companies with high brand recognition could probably convince independent software developers to begin developing application programs even prior to the introduction of their operating systems.

V.9. CONTU seems to have given short shrift to these considerations. CONTU reasoned that there must be adequate legal protection for computer software if authors are to have the incentive to create and disseminate their works. CONTU posited that computer programs would be disseminated only if:

> (1) the creator may recover all of its costs plus a fair profit on the first sale of the work, thus leaving it unconcerned about the later publication of the work; or

(2) the creator may spread its costs over multiple copies of the work with some form of protection against unauthorized duplication of the work; or

(3) the creator's costs are borne by another, as, for example, when the government or a foundation offers prizes or awards; or

(4) the creator is indifferent to cost and donates the work to the public.

CONTU Final Report, supra note 1 at 11. CONTU, however, overlooked the ability of industry leaders to capitalize on their reputations. Given the importance of network externalities in the computer market, this factor is significant. IBM's meteoric rise to leadership in the microcomputer market was no doubt assisted by its reputation in other computer and business equipment markets.

V.10. See supra text accompanying notes III.30-III.31.

V.11. Such agreements. however, may raise antitrust problems. Cf. Ordover & Willig, supra note III.33 (recommending special treatment for high technology joint ventures).

V.12. Leading firms in the computer industry have formed consortia to undertake research and development on projects considered too basic, long-term, or speculative for firms to undertake individually. Microelectronics and Computer Technology Corporation -- an eighteen firm joint venture brought together by Control Data Corporation -- has a $60 million budget to work on ways to improve software development processes. <u>See</u> Software Trade Study, <u>supra</u> note II.6 at 58.

V.13. <u>See</u> CBEMA Data Book, <u>supra</u> note 6 at 32.

V.14. In 1983, approximately 7% of the National Science Foundation budget and 12% of the Department of Defense basic research budget went toward research in mathematics and computer science. <u>See</u> Peck, <u>supra</u> note III.15 at 10, 17.

V.15. <u>See</u> Software Trade Study, <u>supra</u> note II.6 at 57 (Software Technology for Adaptable and Reliable Systems project ($5 million for 1984).

V.16. <u>See</u> <u>id</u>. (Strategic Computing project ($750 million to $1 billion over the period 1984-1994)).

V.17. <u>See</u> <u>id</u>.

V.18. <u>See</u> Software Trade Study, <u>supra</u> note II.6 at 60.

V.19. Minicomputers & Small Business Computers: A Market Survey 42 (Marketing Services Department, The Wall Street Journal, 1979).

V.20. See Samuelson, supra note III.37 at 727-53.

V.21. See M. Nimmer, Nimmer on Copyright section 1.10 [B][2] (1985).

V.22. See Case Comment, The Application of Copyright Law to Computer Operating Systems: Apple Computer, Inc. v. Franklin Computer Corp., 17 Conn. L. Rev. 665, 700 (1985) (noting that copyright might not provide adequate protection against piracy because it only proscribes copying: "The value of the operating system lies in its usefulness as well as its form. Pirates careful not to copy slavishly may legally appropriate return on the original programmer's creative effort, regardless of copyright, by imitating the logical process embodied in the original programs.").

V.23. See Stern, "Idea" Swallows "Expression," or a Left-Handed Way to Say That Second Comers Should Build Their Own Highways to the Market, 2 Computer L. Rep. 380, 380-82 (1983); Note, supra note 9 at 122; Heinlein, Software Lock-in and Antitrust Tying Arrangements: The Lessons of Data General, 5 Computer/Law J. 329 (1985).

V.24. See supra text accompanying notes III.30-III.31.

V.25. This lock-in theory was the basis for the Ninth Circuit's ruling in Digidyne Corp. v. Data General Corp., 734 F.2d 1336 (1984). See infra text accompanying note V.47.

V.26. See B. Kelly & D. Grimes, supra note V.3 at xx.

V.27. See Rubin & Strehlo, supra note V.3 at 65 (quoting Steven Jobs, then-chairman of Apple Computer, as stating "IBM is going to come out with a proprietary version of PC-DOS that other companies won't be able to buy from Microsoft").

V.28. See supra text accompanying note III.22.

V.29. See IBM vs. The Clones, Business Week 62, 68 (July 26, 1986).

V.30. See generally M. Nimmer, supra note V.21 at _______.

V.31. See B. Kelly & D. Grimes, supra note V.3 at xx.

V.32. CONTU Final Report, supra note 1 at 50.

V.33. Apple Computer v. Franklin Computer, 714 F.2d 1240, 1249 (1983). See Stern, supra note V.23 at 382.

V.34. 714 F.2d at 1253.

V.35. Id.

V.36. See supra text accompanying notes III.20-III.21.

V.37. See Brock, supra note III.19.

V.38. Id.

V.39. See Wilson, supra note 11 at 58; see also Pournelle, supra note 11 at 81 (noting the great confusion among microcomputer operating systems).

V.40. Cf. Note, Copyrighting Object Code: Applying Old Legal Tools to New Technologies, 2 Computer/Law J. 421 (1983) (proposing the development of a hybrid "idea copyright" to enable creators of programs to protect both the expression and ideas of computer programs).

V.41. See supra Part I.

V.42. In re Pardo, 684 F.2d 912 (C.C.P.A. 1982); In re Bradley, 600 F.2d 807 (CCPA 1979), aff'd sub nom. Diamond v. Bradley, 450 U.S. 381 (1981). See Davidson, Protecting

Computer Software: A Comprehensive Analysis, 1983 Ariz. State L.J. 611, 674.

V.43. Congress has recently followed a sui generis approach in designing legal protection for semiconductor chips. See supra text accompanying notes IV.64-IV.68. Other commentators have also urged Congress to create a hybrid form of legal protection for computer software. See Samuelson, Creating a New Kind of Intellectual Property: Applying the Lessons of the Chip Law to Computer Programs, 70 Minn. L. Rev. 471, 507 (1985); Karjala, supra note III.36 at 61-81; Stern, The Case of the Purloined Object Code: Can it Be Solved? Part 2, Byte 210, 222 (Oct. 1982); Galbi, Proposal for New Legislation to Protect Computer Programming, 17 Bull. Copyright Soc'y 280 (1970).

V.44. See Economics of Property Rights, supra note 13 at IV-1 - IV-58 (analyzing the optimal duration of legal protection for computer software); Karjala, supra note III.36 at 63 (noting that the effect of legal protection on compatibility should be considered before IBM is granted a seventy-five year period of protection for its Basic Input/Output System) and 67-68 (warning that too long a period of legal protection for computer software might inhibit development of software and hardware).

V.45. <u>See</u> Samuelson, <u>supra</u> note V.43 at 522.

V.46. <u>Cf</u>. Karjala, <u>supra</u> note III.36 at 68 (noting that Japan is carefully considering the desirability of compulsory licensing in a scheme of legal protection for computer software; Note, <u>supra</u> note 1 (proposing compulsory licensing for "idea copyrights" for computer programs).

V.47. 734 F.2d 1336 (1984).

V.48. The court accepted the plaintiffs' proof of market power on the basis of software "lock-in." Software lock-in occurs when a computer user develops or purchases application software designed to run on a particular operating system. This installed base locks the consumer into the hardware products of the owner of that operating system if competitors cannot gain access to the operating system and the costs of converting software to run on different operating systems are high. <u>But</u> <u>see</u> Heinlein, <u>supra</u> note V.23 at 337, 342-44 (1985) (suggesting that conversion costs might not be so high as to justify a finding of market power).

V.49. <u>Cf</u>. <u>Associated Press v. United States</u>, 326 U.S. 1 (1945) (requiring new service formed by over 1,200 newspapers to grant competing papers access to the service);

United States v. Terminal Railroad Ass'n, 224 U.S. 383 (1912) (requiring 13 railroads that controlled access to the only railway bridge into St. Louis to share it with competitors); see generally Note, Unplugging the Bottleneck: A New Essential Facility Doctrine, 83 Colum. L. Rev. 441 (1983) (describing the essential facilities doctrine in antitrust law).

V.50. See generally, Stern, Determining Liability for Infringement of Mask Work Rights Under the Semiconductor Chip Protection Act, 70 Minn. L. Rev. 271, 359 (1985) (describing the legal standard for determining a reasonable royalty).

V.51. See supra note V.3.

Footnotes for Part VI

VI.1. See Keefe & Mahn, Protecting Software; Is It Worth All the Trouble? 62 A.B.A. J. 906 (1976).

VI.2. See Software Trade Study, supra note II.6 at 12.

VI.3. See Sacks, To Copy-Protect or Not to Copy-Protect?, Popular Computing 73 (Oct. 1985).

VI.4. See Software Trade Study supra note II.6 at 12, 14.

VI.5. The rapid advancement of software, however, means that software producers realize most of the return to research and development within a short time after they market their products. See Senecker & Pearl, Software To Go, Forbes Mag. 93, 94 (June 20, 1983).

VI.6. See Antonoff, The New Spreadsheets, Personal Computing 107 (Nov. 1985).

VI.7. See Sacks, supra note VI.3 at 73; Gilburne & Johnston, supra note IV.53 at 226-27.

VI.8. More than 70% of those surveyed considered documentation and ease of use to be "very important." The next most important criterion was "features/performance," garnering a 60.6% percent very important rating. See 1984 Software User Survey, Software News (1984) (cited in Software Trade Study, supra note II.6 at 50).

VI.9. See M. Breslow, A. Ferguson, & L. Haverkamp, An Analysis of Computer and Photocopying Issues From the Point of View of the General Public and the Ultimate Consumer, 121 (1978) (study prepared for CONTU).

VI.10. Karjala, supra note III.36 at 67-68 (noting further that "many new programs are developed through improvements or additions to existing programs. If the improvement is determined to be an adaption, its use might be enjoined [under traditional Copyright protection], resulting in impediments to program development as well as duplicative investments in programs that accomplish similar functions." (footnote omitted)).

VI.11. See Miller, Software Integration, Popular Computing 106 (Dec. 1983) (noting that "the biggest names in software are now introducing sophisticated products that integrate standard bisiness needs to permit data transfer, multiprocessing, and greater ease of use).

VI.12. See supra text accompanying note III.17.

VI.13. _Cf_. _Interview: John Sculley, Apple Computer Inc., President and Chairman -- On Fitting in the IBM World of Computing_ 145 Personal Computing (April 1986) (pointing out that "[i]t's becoming apparent that the real cost is not the hardware or even the software. The real cost is training the user.").

VI.14. _See_ _e.g._, Brock, _supra_ note III.19 at 82-85 (describing IBM's unwillingness to particpate in voluntary effort to establish COBOL as the standard business programming language).

VI.15. _See_ _supra_ note V.43 and accompanying text.

VI.16. _See_ note VI.5.

VI.17. _See_ _supra_ text accompanying note V.44.

VI.18. _See_ Samuelson, _supra_ note V.43 at 496.

VI.19. _Cf_. _id_. at 524-25 (recommending similar provision).

Footnotes for Part VII

VII.1. Note, The Expansion of the Berne Convention and the Universal Copyright Convention to Protect Computer Software and Future Itellectual Property, 11 Brooklyn J. Int'l L. 284, 399 n.88 (1985).

VII.2. See id.