Subject-matter Patentability and Effective Protection of Computer Programs

by

Dae-Hwan Koo

Submitted for the Degree of Ph.D

July 2002

Department of Law
University of Sheffield

CONTENTS

Subject-matter Patentability and Effective Protection of Computer Programs

- *Summary* -

Daehwan Koo

Computer programs can now be protected by patents in the EPO, the US, and Japan. Patents can also be obtained for software implemented business methods in the US. This study highlights the problems of the patent system in protecting computer programs in general, and business methods in particular. One of the main problems is in relation to the economics of software innovation. There have been many disputes on the proper level of protection for software-related inventions to optimize innovation. Another problem relates to the criteria of software patentability. Patentability criteria are different in national patent offices around the world. This can lead to disputes between nations and cause complicated legal problems.

Recognizing these issues, this study examines the fundamental question of whether or not protecting software by existing legal regimes is optimum and desirable in the light of an economic perspective. This discussion reveals a number of disadvantages of the existing legal regimes and leads us to investigate possible alternatives to protect computer programs appropriately.

Thus, this study examines the basic structures and features of the alternative systems, which include a 'Market-Oriented Legal Regime', a 'Compensatory Liability Regime', 'Utility Models', 'Direct Protection of Innovation' and 'Self-Help'. Evaluation of the alternative systems through economic perspectives on the basis of the characteristics of modern software development vindicates that the Direct Protection of Innovation proposed by Kingston and Kronz is the most appropriate form of protection for computer programs. To evaluate this more exactly, the development of software is discussed. This study also investigates the main issues that should be considered in introducing the direct protection system to protect software at the international level.

Abbreviations

APIs: Application programming interfaces

BMI: Business method invention

BMIs: Business method inventions

BMP: Business method patent

BMPs: Business method patents

CAFC: Court of Appeals of the Federal Circuit

CHI: Computer-human interaction

CPU: Central processing unit

DOS: Disk operating system

EPO: European Patent Office

EPO Guidelines: The Guidelines for the Examination in the EPO

European Commission: Commission of the European Communities

FWA test: The Freeman-Walter-Abele test

IP: Intellectual property

IPR: Intellectual property right

IPRs: Intellectual property rights

ISVs: Independent software vendors

JPO: Japanese Patent Office

KIPO: Korea Industrial Property Office

LANs: Local area networks

OS: Operating system

OSs: Operating systems

PC: Personal computer

PCs: Personal computers

PCT: Patent Cooperation Treaty

RAM: Random access memory

ROM: Read only memory

R&D: Research and development

SMEs: Small and medium enterprises

SCPA: Semiconductor Chip Protection Act

Technical Boards: The Technical Boards of Appeal of the EPO

UK: United Kingdom

UKPO: UK Patent Office

US: United States of America

USPTO: United Sates Patent and Trademark Office

WIPO: World Intellectual Property Office

35 U.S.C.: The Title 35 Patents of United States Code

CHAPTER 1

Introduction

The main concern of this study is the necessity of protecting software-related inventions effectively. It is set against the backdrop of the rapid development of electronic commerce (e-commerce) related technologies and the sharp increase in the number of resulting patent applications. In spite of many debates,[1] computer programs are now protected by patents in the European Patent Office (EPO),[2] the United States of America (US), and Japan. Patents can also be obtained for software implemented business methods in the US.

This study highlights the problems of the patent system in protecting computer programs in general, and business methods in particular. One of the main problems is regarding the economics of software innovation. In relation to encouraging innovation, there have been many disputes on the proper level of protection for software-related inventions. Proponents for software patenting argue that patent protection will encourage, and would have encouraged, more innovation in the software industry. Opponents maintain that software patenting will stifle innovation, because the characteristics of software are basically different from those of the inventions of the old industrial society, e.g. mechanics and civil engineering.

Another main problem is related to the criteria of software patentability.

---

[1] In the US, software was generally considered excluded from patent protection until the late 1980s' court decisions.

[2] The EPO grants European patents for the contracting states to the European Patent Convention (EPC), signed on 5 October 1973 and entered into force on 7 October 1977. The EPO was set up by the contracting states to the EPC with the aim of strengthening cooperation between the countries in the protection of inventions. The EPC makes it possible to obtain such protection in several or all of the states by a single patent application, and establishes standard rules governing the treatment of patents granted by this procedure. By filing a single application in one of the three official languages (English, French and German) it is possible to obtain patent protection in some or all of the 20 EPC contracting states. See www.european-patent-office.org/epo/pubs/brochure/general/e/extendhorizon_e.htm accessed 8 June 2002. With regard to the European Commission, see this study, 3.3 [3] Current Situation in the EPO.

Patentability criteria are different in national patent offices around the world. An invention may be patented in the US, but be rejected in the EPO. The US appears to be more generous in granting patents for software-related inventions, especially for business method inventions (BMIs), than the EPO and Japan.[3] This trend may cause the US to monopolize computer-related patents, and this can lead to disputes between nations and cause complicated legal problems. Unless the patentability of all national systems is harmonized, these kinds of problems could deteriorate.

Recognizing these issues, this study examines the fundamental question of whether or not protecting software by existing legal regimes (e.g. patent, copyright and trade secrecy) is optimum and desirable in the light of economic perspective. This discussion reveals a number of disadvantages of the existing legal regimes and leads us to investigate possible alternatives to protect computer programs appropriately.

Thus, this study examines the basic structures and features of the alternative systems. Evaluation of the alternative systems through economic perspectives on the basis of the characteristics of modern software development vindicates that the Direct Protection of Innovation proposed by Kingston and Kronz is the most appropriate form of protection for computer programs. To evaluate this more exactly, the development of software is discussed. This study also investigates main issues that should be considered in introducing the direct protection system to protect software.

This is not the first proposal of a *sui generis* approach to legal protection of software. The World Intellectual Property Organization (WIPO) and the government of Japan proposed a *sui generis* form of legal protection for computer programs.[4] Pamela

---

[3] In a recent study, more than 400 business methods were found to have been filed at the EPO during 1996-1999. Only 5 of them have been granted. See Michal Likhovski, Michal Spence, and Michael Molineaux, *The First Mover Monopoly,* Oxford Intellectual Property Research Centre and Olswang Solicitors, October 2000. See www.olswang.com/scripts/patent_study.pl visited 27 October 2000.

[4] See WIPO, *Model Provisions on the Protection of Computer Software*, 14 Copyright 6, 1213 (1978).

Samuelson, Randall Davis, Mitchell D. Kapor, and J.H. Reichman suggested A Market-Oriented Regime in 1994.[5] They offered a basis on which a complementary or substitute legal regime might be constructed. The market-oriented regime will be discussed in detail in Chapter 5 which deals with alternative proposals. Based on the proposition of *A Manifesto*, Mark Aaron Paley suggested *A Model Software Petite Patent Act* in 1996.[6] *A Model Software* provides a new weaker petite patent tailored to match the nature of software and its market. It has characteristics such as the protection of software that is used in commerce only, central claims, first-to-file method, compulsory licensing for commercial users (and free licence for non-commercial users), blanket licensing[7] and so on. *A Model Software* differs from the market-oriented regime in that it defines what is protectable. Paley defines the term "algorithm" broadly. The broad definition may contain all five of the software entities, (i.e. program code, program compilation, subcompilation, algorithms, and features), and protect them with a single scheme. Paley proposes a world unified Algorithm Office where filings may be made to grant protection in every country.[8]

There are five main purposes in this present study. First, it is to compare the patentability of software-related inventions in the EPO, the US, and Japan. The comparison reveals differences in patentability depending on jurisdictions and

---

The UN asked WIPO to prepare a study on the appropriate form of the legal protection for programs. WIPO prepared twice a draft treaty to constitute an International Union for the Protection of Computer Software. The first was presented in 1976. It provided registration and deposit of the software to be protected. See *Draft Agreement for the Protection of Computer Software and its International Deposit*, WIPO Document AGCP/NGO/III/3, April 7, 1976. The second draft treaty was presented in 1983. It did not include any register. Draft treaty for the Protection of Computer Software, WIPO Document LPSC/II/6, 17 June 1983. See http://swpat.ffii.org/penmi/linuxtag-2001/jh/indexen.html.

[5] See Pamela Samuelson, Randall Davis, Mitchell D. Kapor, and J.H. Reichman, *A Manifesto Concerning The Legal Protection of Computer Programs*, 94 Colum. L. Rev.2308 (1994) (afterwards, Samuelson, *A Manifesto*).

[6] Mark Aaron Paley, *A Model Software Petite Patent Act*, 30 January 1996 (afterwards, *A Model Software*). Available at http://members.aol.com/paleymark/ModelAct.htm accessed 1 June 2001.

[7] Blanket licensing lowers costs substantially, compared to individual licensing. It benefits sellers and buyers.

[8] Paley, *A Model Software*, p. 90.

inconsistency in each jurisdiction. Second, it is to answer to the question of whether or not software patenting in general, and business method patenting in particular, is desirable from an economic perspective. Third, it is to find the most appropriate form of protection for software by evaluating alternative proposals in the light of the characteristics of software and its market as well as modern software development. Fourth, it is to define the subject-matter of the alternative protection systems. Fifth, it is to provide suggestions that should be considered in order to introduce the new regime at the international level. This study will suggest a new approach to the issue of protecting computer programs and business methods based on the Direct Protection of Innovation proposed by Kingston and Kronz. Thus, the real novelty of this study would exist in the application of the Direct Protection of Innovation to the problems of software protection thrown up by existing regimes. Another novel aspect of this study would be found in the effort to define the subject-matter of the alternative proposals and evaluate the alternatives in the light of software development and the defined subject-matter.

This study begins in Chapter 2 by investigating the development of software which highlights the characteristics of the development of modern software and its market. Modern software development is largely characterized as sequential and incremental. Software innovations are now constrained by the need to abide by a standard. In addition, they are vulnerable to copiers. These characteristics provide the basis of evaluation in Chapter 7.

Chapter 3 explains the main issues in computer-related inventions in order to address the issue of subject-matter patentability for computer-related inventions. This chapter, for convenience of explanation, seeks to define business methods and business method patents (BMPs). This chapter tells the story of how standards of subject-matter

patentability of computer-related inventions have developed in these three jurisdictions. This chapter examines patent protection for computer-related inventions in these jurisdictions by reviewing cases and regulations. This chapter also explains theories concerning software-related inventions and the arguments about patent and copyright protection for software-related inventions.

Chapter 4 reviews the historical and legal background to BMPs by dealing with information technology development and pro-patent policy trends. This chapter explains subject-matter patentability of BMPs and tells the story of how BMPs have been introduced and developed. This chapter summarizes software patenting of each of these jurisdictions.

Chapter 5 deals with the issue as to whether software patenting is desirable to encourage innovation. After reviewing the desirability of general software patenting, the desirability of business method patenting is reviewed more specifically. This chapter also examines the desirability of copyright protection for software from an economic perspective. According to the economic analysis, the existing regimes (i.e. patent, copyright law and trade secrecy) do not provide appropriate protection for software innovations. It is necessary to provide new legal protection for these small innovations.

Thus, Chapter 6 provides alternative proposals which include 'A Market-Oriented Legal Regime', 'Compensatory Liability Regimes', 'Utility Models', 'Direct Protection of Innovation' and 'Self-Help Systems'. In this chapter, this study tries to define the subject-matter that these alternative proposals are seeking to protect.

Chapter 7 evaluates the alternative proposals in the light of the characteristics of modern software development (Chapter 2) and the economic analysis (Chapter 4), and it tries to find out the most appropriate form of protection for software at the present day.

It should be considered whether the defined subject-matter of each of the alternative proposals matches with modern software innovations. According to the evaluation revealed in this chapter, the Direct Protection of Innovation proposed by Kingston and Kronz is the most appropriate form of protection for software because it solves the most serious problems of the existing regimes and has many advantages such as familiarity and feasibility. This chapter also provides suggestions of how the Direct Protection of Innovation could be introduced at the international level.

CHAPTER 2

The Development of computer software and its market

2.1 Introduction

A computer program is defined as a set of statements or instructions to be used directly or indirectly in a computer in order to bring about a certain result.[1] WIPO defined the term "computer program" as:

> *A set of instructions capable, when incorporated in a machine readable medium, of causing a machine having information processing capabilities to indicate, perform or achieve a particular function, task or result.*[2]

Software provides the instructions that enable a computer to perform tasks that serve the users' needs. Software can control either the relatively simple operation of a clock or the highly complex control of an airplane. Modern spreadsheet and word-processing programs have over a million lines of instructions or code. Software has become very complex in design during the process of making computers more effective and easy to use.[3]

Software development has evolved from a significantly hardware-constrained activity to a highly flexible and complicated field of engineering.[4] The ultimate goal of computer programming is the design of a functional work.[5]

---

[1] The US Copyright Law at Section 101.
[2] WIPO, *Model provisions on the protection of computer software*, 1978.
[3] The engineering nature of software design is discussed in this study, 5.2 Characteristics of Software. See also *A Manifesto.*
[4] www.law.berkeley.edu/institutes/bclt/pubs/ipnta/appenb.pdf accessed 13 October 2001.
[5] Copyright protection for software has become inadequate because it is insufficient to protect the functions i.e. the ideas embodied in a program. Patent protection for software is increasing. A patent may claim a system or a method for accomplishing a particular function. See this study, 5.4 Desirability of Software Patenting.

In order to find out the most appropriate sort of protection for software innovations, it is necessary to investigate the characteristics of the development of software and its market. This is because the appropriate protection of software would reflect the characteristics of the development of software and its market, and because it would encourage the innovation of software technology. This section of this study describes the early history of computing and the principal methods involved in modern software development, especially the development of relatively complex commercial operating systems[6] (OSs) and application programs.[7]

2.2 *History of Computer Software*

The first computers appeared during the 2nd World War and were devices for carrying out the high-speed calculations that were necessary for code breaking.[8] The computer was planned from the beginning as an *artificial brain* that *thinks* in the way of humans and executes logic.

In 1937, a 24 year old British scientist called Turing incorporated the concept of algorithms into Boolean logic[9] and theoretically completed the computer.[10] Turing presented the theory of the universal algorithm machine which could perform various

---

[6] OSs form the interface between the very basic and difficult to use machine language of the computer and the more human-like abstract languages with which most application programmers work. See http://www.law.asu.edu/HomePages/Karjala/Articles/JurimetricsFall1987.html#FN;Fa accessed 21 August 2001. Dennis S. Karjala, *Copyright, Computer Software, and the New Protectionism,* 28 Jurimetrics Journal 33 (Autumn 1987).

[7] Application software covers all programs whose purpose is to solve the computer user's problems. Examples of such problem-oriented software are word-processing programs, calculation software, presentation software, process controlling software, accounting software and so on. See http://www.diw-berlin.de accessed 13 October 2001.

[8] JPO, Asia-Pacific Industrial Property Center, JIII, *Practical procedures for prosecuting software patents,* 1999 (afterwards, JPO, *Practical procedures*).

[9] Boolean algebra is used for designing so-called logic networks for digital computers. 1996 Encyclopaedia Britannica, Inc.

[10] JPO, *Practical procedures.*

kinds of computations using only two types of symbol. The theory of a computer was introduced before an actual computer was created. This is the reason why the computer was developed as an artificial brain or a device carrying out logical computation in a way similar to human beings. Software possesses an essentially different aspect from machinery, electrics and chemistry. This different aspect is that software is an execution of complex logic. The execution of complex logic is a skill exclusive to human beings and is called a *mental step*. Mathematics is vital in the invention of software, which is composed mainly of *abstract algorithms*.[11]

A second form of computer was developed in the field of device control.[12] Control systems for all types of machinery were developed for mechanical control and electric control. In about 1930, there was research into an automatic telephone exchange machine using relay circuits. Since the circuits were extremely complex, it was difficult to ascertain accurate operations. In 1937, 22-year-old Claude Shannon applied Boolean theory to electrical switch circuits in his doctoral thesis. Due to this study, electric circuits were introduced into the field of sequential control circuits. Initial relays contained the sequential control circuits and they were replaced by semiconductors later.

The critical breakthrough in computer history was the exploitation of electrical impulses to process information.[13] In 1939, Professor John Vincent Atanasoff and his graduate student Clifford Berry developed the first electronic calculating machine. This computer could solve relatively complicated physics computations. They developed a more sophisticated version, the ABC (Atanasoff Berry Computer) in 1942. In 1946, Dr.

---

[11] Because of the characteristics of mental step and abstract algorithms, software was not regarded as patentable. See this study, 3.2 [a] The Mathematical Algorithm Exception, and 3.2 [b] Mental Step Doctrine. An algorithm is a step-by-step set of instructions designed to accomplish some data processing task. See also http://papers.ssrn.com/sol3/papers.cfm?abstract_id=239903 accessed 16 October 2001.

[12] This would be the same case of *control for hardware resources, or processing with respect to the control* which is regarded as utilizing natural laws in Japan. See this study, 3.4 [2] [a] Basic Concept of Utilizing Laws of Nature; JPO, *Practical procedures*.

[13] www.law.berkeley.edu/institutes/bclt/pubs/ipnta/appenb.pdf.

John Mauchly and J. Presper Eckert completed the Electronic Numerical Integrator and Computer (ENIAC). ENIAC was about 1,000 times faster than the previous generation of relay computers. It is commonly accepted as the first successful high-speed electronic digital computer (EDC). This computer occupied 15,000 square feet, weighed 30 tons. It operated in decimal rather than binary and therefore needed 10 vacuum tubes to represent a single digit.

Fascinated by the success of ENIAC, the mathematician John Von Neumann undertook an abstract study of computation.[14] The study showed that a computer should have a very simple, fixed physical structure, and that it could execute any kind of computation through a proper programmed control without any change in the unit itself. Neumann contributed to the awareness of how practical and fast computers could be organized and built. These ideas, which are referred to as the stored-program technique, became essential for future generations of high-speed digital computers and were universally adopted.

Advances in the 1950s

Innovative computer technologists have existed since the 1950s. Early in the 1950s two important engineering discoveries were made. These discoveries were the magnetic core memory and the Transistor Circuit Element. These technical discoveries quickly led to new models of digital computers. RAM[15] capacities have increased from 8,000 to 64,000 words in commercially available machines by the 1960s. Since it was very expensive to purchase or to rent these machines, batch processing was developed.[16] In

---

[14] http://isu.indstate.edu/welsh/ua/hist-comp.html accessed 6 October 2001.
[15] RAM represents 'random access memory' and ROM, 'read only memory'.
[16] Batch processing, in the 60s and 70s, became the usual mode of operation of mainframe computers. See www.science.uva.nl/faculteit/museum/technotrends.html accessed 6 October 2001.

batch processing, problems are prepared and held ready for computation on a relatively cheap storage medium. When the computer finishes with a problem, it copies the whole problem (program and results) and moves it on one of these peripheral storage units and starts on a new problem.

In the early 1950s, programming languages were developed for the purpose of providing a human interface to the computer and in the beginning relied heavily on the hardware capabilities.[17] Languages were developed firstly as a result of the availability of compiler and translator technology and subsequently as a means of meeting user needs.

In the early part of the decade there was a need for the development of "automatic programming" from the industry.[18] Betty Holberton was probably the first to write a program that generated another program. Building on this idea, Hopper conceived of the compiler, and high level programming languages started to develop. High-level computer languages enabled computer specialists to write programs using coded instructions that resemble human language.

The turning point came in April 1957 when the first FORTRAN (Formula Translator) compiler was released by John Backus and his team at IBM. This system could use a mathematical-like notation. FORTRAN was the prototype system showing the feasibility of concept compiling. The IBM 705[19] used the FORTRAN language. This model became the standard machine for large-scale data processing companies.

Since the 1950s the computer has replaced traditional methods of accounting and

---

[17] www.computer.org/students/looking/spring97/janlee/ accessed 6 October 2001.

[18] Ibid. The 1950s were a decade when the computer industry firstly became commercialized and then turned from a situation restricted to the scientists and mathematician to a situation where the person with the problem was more important than the person with the method of solution.

[19] The IBM 705 was introduced in 1959. See www.law.berkeley.edu/institutes/bclt/pubs/ipnta/appenb.pdf.

record keeping and given rise to a new data processing industry.[20] As a primary means of communication over time and space, it has formed the core of modern information technology. The flexibility provided by programmability greatly enhanced the utility of computers.[21] In the early 1950s, Mauchly and Eckert developed the first *commercial* electronic computer, the Universal Automatic Computer (UNIVAC 1) for Remington-Rand Corporation. Because of the limitations on electronic technology, however, the computing power of the first generation of computers was constrained. Vacuum tubes were bulky, failed frequently, consumed large amounts of energy, and generated considerable heat. This first generation of computers was programmed in binary code. In 1954, IBM introduced its first commercial computer, the IBM 650. IBM made incremental improvements to this technology and emerged as the market leader.

Because computers used binary electronic switches to store and process information, it was necessary to reduce the size of these switches.[22] The second generation of computers replaced vacuum tubes with transistors. Transistors were smaller, required less power and generated less heat. This and other innovations in data storage technology made computers smaller, faster and more reliable. The first scientific computer using transistors was the IBM 7090. Despite of these innovations, computers of this generation remained complicated and expensive because circuits had to be wired by hand.

Advances in the 1960s

Because of the greater computing power and efficiency of computers, companies

---

[20] www.princeton.edu/~mike/articles/hcht/hcht.html accessed 6 October 2001. *The History of Computing in the History of Technology*, Michael S. Mahoney, Annals of the History of Computing 10(1988), 113-125.

[21] www.law.berkeley.edu/institutes/bclt/pubs/ipnta/appenb.pdf.

[22] Ibid.

increasingly wanted to rely on data processing services.[23] Some companies purchased their own computers. IBM's 360 series of mainframe computers emerged during this period as the market leader.[24] These machines used a single machine language. Since companies upgraded their equipment within the 360 series, they could continue to use the same computer programs, resulting in expanded market and increased benefit of owning a computer. This larger market generated greater demand for computer programmers, increased the number of companies providing computer-related services, and encouraged the emergence of an independent software industry.

Computer science began to take shape during the 1960s, as it brought together common concerns from mathematical logic, mathematical linguistics, and numerical analysis. In the 1960s, users in universities and in research laboratories developed two design concepts, the minicomputer and timesharing, which fundamentally changed the way in which computers would be used. Many scientists learned about using computers by programming them directly. They developed small and specialized laboratory computers which were far cheaper than the centralized systems that often cost more than a million dollars.

Timesharing was developed because a large system costs hundreds of dollars per minute to operate. The computer would be idle most of the time while waiting for its user to initiate some action, because humans work very slowly compared with computers. Timesharing was designed to make such kind of direct use efficient by enabling a computer to serve many users simultaneously. With this system, each user sits at a terminal. The computer transfers its attention rapidly from one user to another, performing works as needed. Information is processed so quickly that the computer

---

[23] Ibid.

[24] A mainframe of the mid-1990s consists of more than 50 times as many transistors as the IBM System/360 of the 1960s. 1996 Encyclopaedia Britannica, Inc.

appears to be totally dedicated to each user's job. That is to say, in timesharing the computer processes many jobs in such a rapid succession that each job runs as if the other jobs do not exist. Such operating modes need elaborate executive programs to attend to the administration of the various tasks. After the concept of timesharing had been incorporated, it became apparent that users did not have to be in the same room as the machine, but could communicate with it over communication lines from anyplace in the world.

Thus, although large centralized systems continued to grow for applications such as record keeping and laborious computations for scientific research, the base was laid in the 1960s for new types of computer use and new ways of designing systems to meet specific goals.

During this period, the major computer manufacturers began to offer a range of capabilities and prices, as well as accessories such as consoles, card feeders, page printers, cathode-ray-tube displays and graphing devices. These were widely used in businesses for accounting, payroll, inventory control, ordering supplies and billing. Computers for these uses did not have to be very fast arithmetically and were usually used to access large amounts of records, keeping these up to date. Computer systems were generally sold for the simple uses, such as hospitals (keeping patient records, medications, and treatments given), libraries, and chemical abstracts system (covering nearly all known chemical compounds).

During the 1960s, object-orientation emerged.[25] One of the goals of object-oriented languages[26] is to have flexibility in programming. Instead of considering

---

[25] http://jeffsutherland.com/papers/Rans/OOlanguages.pdf accessed 14 October 2001, *A History of Object-Oriented Programming Languages and their Impact on Program Design and Software Development.*

[26] Object-oriented programming is a programming technique in which a program is written with discrete objects that are self-contained collections of computational procedures and data structures. New programs

procedures and data as being separate, object-orientation unifies them into a group called an object. A minor change in requirement of a program should not demand an entire rewriting. If other programs share any of their features, adaptation of the relevant sections can save time and money. Software production can be expedited by the construction of libraries of classes. The division of programs into objects makes this task relatively simple.

Software languages have improved dramatically since the 1960s, with the introduction of Unix,[27] Lisp,[28] C[29] and many other programming and system languages. These languages supported the decomposing of development work into small modules that permitted teams to work with minimal communication.[30] Software design started to move from art to routinized tasks manipulating standardized modules. The entire cycle of software production, installation and maintenance became factory-like procedures and processes. This suggests that writing programs is an industrial design process.[31]

---

can be written by assembling a set of these predefined, self-contained objects in far shorter time than by writing complete programs from scratch. Recently, object-oriented programming has become extremely popular because of its high programming productivity. C++, which was developed by Bjarue Stroustrup of AT&T Bell Laboratories in the early 1980s, and Objective-C, which was developed by Brad Cox in 1984, are object-oriented versions of C that have gained much popularity. 1996 Encyclopaedia Britannica, Inc.

[27] Unix is a programming language, first developed by AT&T's Bell Laboratories, which became widely disseminated in the development community.

[28] LISP (List Processor) is a language that is powerful in manipulating lists of data or symbols rather than processing numerical data. In this sense, LISP is unique. It requires large memory space and, since it is usually processed by an interpreter, is slow in executing programs. LISP was developed in the late 1950s and early 1960s by a group headed by John McCarthy, then a professor at the Massachusetts Institute of Technology. At that time, LISP was radically different from other languages, such as FORTRAN and ALGOL. Several versions have been developed from the LISP 1.5 introduced by McCarthy; Common LISP, released in 1984, is becoming the de facto standard of LISP. 1996 Encyclopaedia Britannica, Inc.

[29] Although C is considered to be a high-level language, it has many low-level features, such as the ability to directly handle addresses and bits. C is, nonetheless, highly portable. It was developed by Dennis M. Ritchie of AT&T Bell Laboratories in 1972. The operating system UNIX has been written almost exclusively in C; previously, OSs were almost entirely written in assembly or machine code. C has been extensively used on personal and larger computers. 1996 Encyclopaedia Britannica, Inc.

[30] See Bruce Kogut and Anca Meitu, *The Emergence of E-Innovation: Insights from Open Source Software Development*, A Working Paper of the Reginald H. Jones Center, WP00-11, The Wharton School, University of Pennsylvania.

[31] See this study, 5.2 [4] Programs are Industrial Compilations; M.A. Cusumano, *Japan's Software*

In 1965, the Digital Equipment Corporation (DEC) introduced the first minicomputer, the PDP-8 (Programmed Data Processor).[32] This computer was quite small and about one-fourth of the price of typical mainframe computers. Minicomputers greatly widened the market for computers.

In the 1960s, the implementation of timesharing and telecommunication technologies enabled multiple users to access a computer from remote terminals, and computers could process multiple tasks simultaneously. During this period, an OS was begun to use.[33]

Advances in the 1970s

In the 1970s, communication-based computer systems began to grow. Immediate data entry and retrieval of information from remote locations were needed. For example, by airline reservation systems, agents all over the country can check flight availability and reserve seats from their work stations in "real time".[34] Multiple computer systems were linked together into networks, both small and large ones.

The trend during the 1970s was generally moving away from very powerful and single-purpose computers and toward a larger range of applications for cheaper computer systems.[35] Most continuous-process manufacturing (e.g. petroleum refining and electrical-power distribution systems) used computers of smaller capability for controlling and regulating their jobs.

Application languages were now available for controlling a great range of

---

Factories, New York: Oxford University Press, 1991.

[32] www.law.berkeley.edu/institutes/bclt/pubs/ipnta/appenb.pdf.

[33] www.digitalcentury.com/encyclo/update/com_hd.html accessed 30 October 2001. *Computers: History and Development.*

[34] "Real time" is used to refer to the situation when the computer provides immediate action and response.

[35] http://isu.indstate.edu/welsh/ua/hist-comp.html accessed 6 October 2001.

manufacturing processes, for using machine tools with computers, and for many other things. Moreover, a new revolution in computer hardware was under way. LSI (large-scale integration) shrank computer-logic circuitry and components.[36] Many companies, such as Apple Computer and Radio Shack, introduced very successful personal computers (PCs). The first personal computer (PC) was developed in 1975.[37]

By the 1970s, computers incorporated semiconductor chips that are as small as a human fingernail and contain more than 100,000 transistors. As chip technology advanced, the size of computers decreased significantly while their capability increased. Semiconductor chips today can contain many millions of transistors. For the past two decades, the memory capacity of a semiconductor chip has doubled approximately every 18 months.[38] Intel Corporation developed the microprocessor, a chip that contains the entire control unit of a computer in the early 1970s. Very large scale integration (VLSI) technology led to the development of the microcomputer. Microcomputers came to dominate the computer industry by the mid-1980s. Apple Computer greatly expanded the computer market with its Apple II computer system which involved a keyboard, monitor, floppy disk drive and OS. Microcomputer unit sales surpassed minicomputer unit sales in 1976.[39]

On the other hand, antitrust scrutiny by the US government and increasing costs of software development forced IBM to unbundle its hardware from application programs in 1970. This event greatly expanded the business opportunities for independent software vendors (ISVs). The sudden increase of minicomputers in the early 1970s encouraged the growth of ISVs and the shift away from custom programming. The

---

[36] In the 1950s it was realized that scaling down the size of electronic digital computer circuits and parts would increase speed and efficiency.

[37] http://eplu.netgistics.com/PDF/1999-2000/Fall/BUSA541/cham1.pdf accessed 14 October 2001.

[38] www.law.berkeley.edu/institutes/bclt/pubs/ipnta/appenb.pdf.

[39] Ibid.

introduction of the microcomputer in the 1970s significantly changed the software industry. With relatively small investments, computer programmers could develop software for the growing numbers of microcomputer users. From the late 1970s, ISVs began to sell pre-packaged (i.e. non-customized) software products for use on microcomputers. Wordstar, Visicalc,[40] and other independently developed software products dominated the early microcomputer software market.

Developments in the last twenty years

By the early 1980s, IBM developed their version of the PC.[41] IBM chose DOS (Disk Operating System) as the PC OS. In August 1980, Microsoft gained the rights to DOS from Seattle Computer Products for less than $100,000. During the 1990s, Microsoft developed DOS to Windows 3.0, Windows 3.1, Windows for Workgroups, Windows 95, and Windows 98.[42]

In the early 1980s the Japanese government announced a gigantic plan to design and build a new generation of supercomputers. This new generation (the so-called "fifth" generation) was using new technologies in very large integration along with new programming languages. By the late 1980s, some PCs were run by microprocessors that could process about 4,000,000 instructions per second through handling 32 bits of data at a time.

By 1986, sales of microcomputers had reached approximately 4 million units and produced revenues of almost $12 billion, which means that microcomputers gave the largest revenues the computer industry had earned hitherto. The rapid growth of the

---

[40] VisiCalc is the first electronic spreadsheet (computerized accounting program). 1996 Encyclopaedia Britannica, Inc.

[41] http://eplu.netgistics.com/PDF/1999-2000/Fall/BUSA541/cham1.pdf.

[42] Windows 3.1 is DOS based, but Windows 95 is not. *The Dictionary of Computer*, Youngjin Publishing Co., (Korea) 1999.

microcomputer encouraged the development of ISVs. Microcomputer owners were eager to develop different programs. The development cost of software for these machines was relatively low and the product cycles were short. These characteristics of software industry forced the software developers to upgrade their products constantly.[43]

IBM entered the microcomputer market in 1981 with its PC product. The IBM PC utilized an Intel microprocessor (16-bit 8088 chip) and an OS (PC-DOS) licensed from Microsoft.[44] IBM's good reputation in the computer industry and its vast distribution network for computers enabled it to attract customers rapidly for its PC products. Many ISVs and hardware manufacturers developed and marketed software and peripheral products (e.g. printers, monitors) to run on the IBM PC. IBM actively encouraged ISVs and the manufacturers of peripheral equipment to develop products for its PC. On the other hand, IBM included a specialized chip, BIOS,[45] for transferring data within the PC, which hindered other manufactures from offering fully compatible computer systems. This enabled IBM to charge additional prices for its PC products.

The success of the IBM PC encouraged ISVs to develop a wide range of programs to run on the IBM PC. These programs included word-processing and spreadsheet software. For example, Lotus Corporation developed a version of the spreadsheet Visicalc[46] to run on the IBM PC platform. Within a year of its introduction, Lotus 1-2-3 surpassed Visicalc and became the spreadsheet market leader. Its success spurred the sales of a hardware/OS platform, which elevated the importance of owning an IBM PC.

The powerful IBM trademark and the wide availability of software designed to run on

---

[43] www.law.berkeley.edu/institutes/bclt/pubs/ipnta/appenb.pdf.

[44] MS-DOS controlled operations such as disk input and output, keyboard control, video support and many internal functions related to program execution and file maintenance.

[45] The BIOS Chip is the set of essential software routines that test hardware at startup, start the OS, and support transfer of data among hardware devices. It is typically stored in ROM so that it can be executed when the computer is turned on.

[46] Visicalc was originally designed to run on the Apple II.

the IBM/MS-DOS platform placed IBM in a dominant position in the early microcomputer marketplace and greatly encouraged the dissemination of microcomputers. These factors spurred rapid growth of software industry.

Because of the availability of software designed to run on the IBM PC platform, other manufactures tried to develop computer systems that could run the growing supply of IBM-compatible software.[47] Although Microsoft's MS-DOS operating system could be licensed in the market, IBM refused to license its BIOS chip. As a result, other manufactures could not fully imitate the internal operations of the IBM PC and some software designed for the IBM PC did not operate satisfactorily on the computer systems of other manufactures. Consumers accordingly strongly favoured IBM PCs. Other computer companies could not but offer IBM PC compatibility. Computer manufacturers that developed their own platform could not succeed. No serious alternative to the IBM PC/MS-DOS platform survived except the Apple Computer which maintained a niche in the marketplace largely because of its superiority in handling graphics.[48]

By 1984, Compaq developed a BIOS chip that could run software designed for the IBM PC. Pheonix Technologies Ltd. developed a fully IBM PC-compatible ROM and BIOS which was licensed to a broad range of manufactures. Other manufactures entered the market for IBM PC-compatible computer systems. As consumers became increasingly confident that application software designed for the IBM PC would run on the computer systems of other manufactures, these computers eroded IBM's dominance of the marketplace by offering lower prices, wider selection, and additional features. By 1986, a number of manufactures competed in the IBM-compatible/MS-DOS

---

[47] www.law.berkeley.edu/institutes/bclt/pubs/ipnta/appenb.pdf.
[48] Ibid.

marketplace and IBM lost its share of the market significantly. The broad range of software available for the IBM-compatible/MS-DOS platform made MS-DOS the *de facto* OS standard in the industry by the late 1980s. Around this time, Microsoft began to develop the Windows OS platform incorporating a graphical user interface similar to that of Apple. The Windows platform was backward compatible with MS-DOS. Applications designed to operate in the MS-DOS environment could run on the Windows platform as well. Most MS-DOS users as well as new users adopted the Windows platform. The Windows platform has been the dominant platform since the mid-1990s.

IBM became, and remained, the major manufacturer in the computer industry from the 1950s to the early 1980s.[49] During the 1950s and 1960s, IBM and other mainframe manufacturers[50] bundled OS and application software with hardware for the same price. Manufacturers encouraged their customers to share software among themselves. IBM formed and supported a user group named SHARE that served as a clearinghouse[51] for programming information and software for computer users. SHARE distributed computer programs, e.g. libraries of subroutines, algorithms, computer code, and programs written to solve problems. The companies contributing to the information sharing were entitled to borrow the others' works.

As the industry developed and computers became increasingly powerful, versatile and less expensive, the sharing model began to break down. Those companies that had invested substantial resources to develop software became unwilling to share their innovations with others. The introduction of less expensive minicomputers and the

---

[49] Ibid.
[50] E.g. Burroughs, Raytheon, RCA, Honeywell, General Electric, and Remington Rand.
[51] Clearinghouse is an organization that collects and exchanges information on behalf of people or other organizations.

growing versatility and computing capacity of mainframes spurred the independent software industry.[52]

By the late 1980s, Microsoft had become a dominant force in the software industry. Its MS-DOS operating system was installed on the majority of microcomputers. By 1991, Microsoft's OSs were installed on almost 90% of microcomputers in the world.[53] On the top of this success, Microsoft began to bundle its office software products into an office set of products that include Microsoft Word (word-processing software), Microsoft Excel (spreadsheet software), Microsoft Access (database software), and Microsoft Powerpoint (presentation software). Due to this marketing strategy, Microsoft has become the leading seller in each of these product categories.

In 1990s, while hardware vendors controlled large and midrange OSs, Microsoft and Apple controlled desktop OS. The proliferation of PCs created the need for connectivity. PCs were not much good for business unless they could communicate each other. Local area networking technology was developed to tie PCs together. The demand for computers that could store documents and applications to be shared between local area networks (LANs) users increased significantly.

In the past three decades, the performance-price gain of computer hardware has been about 30% per year.[54] Even though software demand is growing, software development has undergone only incremental improvement. Moreover, the cost of software testing and maintenance has greatly increased. In order to secure the reliability of software, efforts have been made to establish standards and regulations.

---

[52] The independent software industry grew rapidly. By 1965, there were approximately 40 to 50 ISVs. There were almost 3,000 vendors by 1968.

[53] www.law.berkeley.edu/institutes/bclt/pubs/ipnta/appenb.pdf.

[54] http://math.ucsd.edu/~fan/rep.pdf accessed 26 October 2001.

2.3 An Introduction to Computer Technology

[1] Computer Software and Hardware

While hardware represents something that cannot easily be changed, software implies ease of change.[55] Software is inherently malleable and flexible.[56] Software code can be expanded, modified and combined to add functionality and bundle features.

The overall concept of a computer is a machine in which hardware and software process information as a single entity. Hardware is an external system to use the information which the computer processes. If the external system is a company accounting system, the computer software employed will be accounting software. If the external system is a car engine, the computer software will be engine control software.

Computer programs allow general-purpose computers to be many different types of machines.[57] When a computer is running a video game, the computer is a videogame machine. When it enables users to write letters, the computer is a word-processing machine. As the *Alappat* court stated, a general-purpose computer becomes a special-purpose computer once it is programmed to perform particular function.[58] Programs can also be written for special-purpose hardware e.g. the semiconductor chip that monitors the functioning of a toaster.

A computer program consists of instruction commands which make a computer

---

[55] JPO, *Practical procedures*. The term "software" was originally used in contrast to "hardware". Hardware means items which have a hard or solid physical presence and a fixed form such as a machine. Hardware refers to the many devices that constitute a computer, which include central processing units (CPU), memory boards, printers, monitors, keyboards, magnetic disks, optical disks and so on.

[56] Steven J. Davis, Jack MacCrisken and Kevin M. Murphy, *Economic Perspectives on Software Design: PC Operating Systems and Platforms*, Working Paper 8411, August 2001 (afterwards, Davis, *Economic Perspectives*). Available at http://www.nber.org/papers/w8411 accessed 14 October 2001.

[57] www.law.berkeley.edu/institutes/bclt/pubs/ipnta/appenb.pdf.

[58] See *In re Alappat*, 33 F.3d 1526 (Fed. Cir. 1994); This study, 3.2 [3] [b] The CAFC's Algorithm-related Cases.

run.[59] Computer programs can be divided into two categories: OS and application programs.[60] Each of these categories provides a different function in computer processing, but is co-related.

An OS program is the program which coordinates all activities on a computer and manages the internal functions of the computer. It provides a basic roadmap for the functionality of the hardware. Its key tasks are to recognize and control input and output functions, manage file directories and the behaviour of the external peripheral functions of the computer.[61] The OS also manages the processing schedule of the microprocessor. It coordinates the reading and writing of data between the internal memory, CPU, and the external devices (e.g. disk drives, keyboard, and printer). Moreover, it facilitates use of application programs. An OS is a software platform that is used to run application programs. In order for an application program to run successfully on a particular OS, the program must be designed to be compatible with the OS. This design is usually based on the use of a common set of application programming interfaces[62] (APIs) which enable the application to communicate with the OS in the same language and can create compatibility. In essence, the OS controls the interactions between the user, the software, and computer itself.

Applications create end-user functionality.[63] Applications enable users to accomplish specific tasks with computers. Applications include word-processors, spreadsheets, bookkeeping, statistical and financial analysis, video game programs,

---

[59] Jurgen Bitzer, *The computer software Industry in East and West: Do Eastern European Countries Need a Specific Science and Technology Policy?* http://www.diw-berlin.de 13 October 2001.

[60] Seth A. Cohen, *To Innovate or Not to Innovate, That is the Question: The Functions, Failures, and Foibles of the Reward Function Theory of Patent Law in Relation to Computer Software Platforms*, 5 Mich. Telecomm. Tech. L. Rev. 1 (1999) (afterwards, Cohen, *To Innovate*). Available at www.mttlr.org/volfive/cohen.pdf accessed 14 October 2001.

[61] Cohen, *To Innovate*, p. 23.

[62] Interface is the connection between systems specified in physical (electrical or mechanical) and/or logical (format, syntax or procedure) terms.

[63] Cohen, *To Innovate*, p. 24.

design programs and so on. Applications are essentially high-level sets of commands that use the underlying computer processing ability to serve specific functions. All applications contain APIs which communicate with the software platform. Without APIs, applications will not initialize or function with the software platform.

The process of adapting an application program designed to run on one OS to another OS is often complex and time-consuming work.[64] Software developers have recently developed programs that run on more than one OS. Nonetheless, the problem of compatibility between application programs and OSs is a major concern in the computer software industry.

The line between OSs and application programs may not be distinct when application programs are integrated into an OSs program. Microsoft has integrated many add-on features such as compression software and basic word-processing into its OS. By purchasing a bundled product that includes both an OS and certain applications, consumers could be benefited. However, Microsoft's bundling has eliminated a competitive market for such add-on products. Microsoft's decision to incorporate its Internet Explorer web browsing software into the Windows OS contributed to the US Justice Department's decision to charge Microsoft with antitrust violations in the late 1990s.[65]

[2] Computer Networks

Easily accessible means of communication and cheap hardware have only been

---

64 www.law.berkeley.edu/institutes/bclt/pubs/ipnta/appenb.pdf.
65 Ibid.

available for the last ten years.[66] Because early computers were self-contained machines,[67] to move data from one computer to another, one had to take the output of one computer, and input it physically into the new computer. The Internet[68] solved this communication problem.

Computer engineers recognized that great benefits could be gained from the networking of computers. Networking has been used in specialized computing environments since the late 1960s. As of June 1999, more than 171 million people around the world had Internet access.[69] Networking technology enables computer users to exchange information. It also reduces the time and cost of transferring information regardless of physical location of computers. Physical location is now irrelevant as long as the Internet is available. Networking enables computers to work together to achieve certain tasks that would take much longer to do alone.

Networking computers requires some basic technologies. Computers must be connected one another physically or virtually,[70] and when one computer sends data, the other must be able to process it. The development of standard protocols for exchanging information has encouraged the growth of networking. Thus, standardization and interoperability issues arise in the context of computer networking.[71]

---

[66] http://eplu.netgistics.com/PDF/1999-2000/Fall/BUSA541/cham1.pdf.

[67] www.law.berkeley.edu/institutes/bclt/pubs/ipnta/appenb.pdf.

[68] The nature of the Internet is discussed in this study, 5.3 [2] The Nature of the Internet. The Internet is a network connecting many computer networks and based on a common addressing system and communications protocol called TCP/IP (Transmission Control Protocol/Internet Protocol). From its creation in 1983 it grew rapidly beyond its largely academic origin into an increasingly commercial and popular medium. 1996 Encyclopaedia Britannica, Inc.

[69] www.law.berkeley.edu/institutes/bclt/pubs/ipnta/appenb.pdf. By 1996, more than 9 million host computers were connected to the Internet. Computer users can access to the Internet through Internet Service Providers (ISPs) such as America Online (AOL) and Microsoft Network (MSN).

[70] If the computers are physically close to one another, connection can be achieved by stringing a cable between the computers. If the computers are physically distant, networking requires that they be connected by telephone lines or by some form of wireless communication.

[71] www.law.berkeley.edu/institutes/bclt/pubs/ipnta/appenb.pdf.

Local area networks

Many organizations find it useful to create LANs to link computers so that their members can share information. LANs enable the members to communicate, exchange files, and share information. Because the computers in a LAN are linked together, it is possible to store files in a single central location and allow any computer to access it at any time. Thus, LANs make it possible to specialize or divide tasks among computers.

LANs also facilitate group projects. Workgroups can change the same document simultaneously over the network. This is a particular advantage for companies e.g. sales companies, airlines, hotels and banks which rely on large information databases that must be frequently updated.

Large-scale public networks

In the 1970s, the Defence Advanced Research Projects Agency (DARPA) worked with a number of university research groups such as MIT, Stanford, and Carnegie-Mellon on military projects.[72] Researchers studied the issue of connecting their computers scattered around the US to enhance memory storage and timesharing capabilities. An important design objective of this system was that it should not be vulnerable to breaking down in the event of a nuclear attack or other widespread disruption of telecommunications systems. Computer programs are now distributed over LANs or the Internet. A communications network was developed to facilitate easier communications among the participants. This network called ARPANET was the predecessor of today's Internet. Internet protocol, TCP/IP,[73] was developed in the late 1970s. The Internet was a largely decentralized system utilizing a common communication standard.

---

[72] http://eplu.netgistics.com/PDF/1999-2000/Fall/BUSA541/cham1.pdf.

[73] TCP/IP is *de facto* standard protocol for the interconnection computers. See www.science.uva.nl/faculteit/museum/technotrends.html accessed 6 October 2001.

As computing has shifted from a standalone activity to one conducted over nétworks in the 1990s, the way that computers function has changed as well. Most computers do not operate in self-contained way. They work with others. Companies such as sales companies, hotels and banks began to learn how to use the Internet and develop new business methods using the Internet.

2.4 Software Development

Innovation in software development is typically cumulative and incremental.[74] Programmers commonly adopt software design elements by looking around for examples or remembering other programs. While innovation in program design occasionally rises to the level of invention, generally it does not.[75] It is the product of the skilled use of know-how to solve industrial design tasks. Program development requires skilled effort and applied know-how comparable to other engineering disciplines.

The products of software engineering generally contain admixtures of old and new elements.[76] The innovation in such programs may exist in the way that the known elements have been combined in a new and efficient manner. It may also lie in the combination of new elements with well-known elements in order to achieve the same result in a new way. When we define programs as "industrial compilations of applied know-how", the definition is based on the recognition that software engineering

---

[74] Direct protection of innovation by Kingston and Kronz considers incremental innovation. See this study, 6.4 Direct Protection of Innovation, and 5.2 Characteristics of Software.

[75] This is one of the reasons that patent protection of software is inappropriate. When innovation is sequential and complementary, imitation becomes a spur to innovation and strong patents become an impediment. Patenting incremental innovations, which are not inventive, would not be appropriate for the economic goals of patent system, since a patent is given for a substantial contribution to the art. See this study, 5.4 [2] Cons for Software Patents.

[76] See *A Manifesto*.

involves the re-use of known elements.

Although new programs are designed and written, a fundamental fact about computer programs is that they are functional.[77] Computer programs are designed to accomplish certain tasks and cause certain behaviour. (In this respect, software protection by copyright has difficulty, because copyright does not protect idea, procedure, process, system and method of operation.)[78] Therefore, programs are judged largely on how well they perform the tasks they have been programmed to accomplish. The concept of efficiency in this context may mean the following: (1) code efficiency (maximizing the processing speed); (2) memory efficiency (minimizing the amount of memory needed to accomplish the desired tasks); (3) input/output efficiency (maximizing the quality and speed of information transmission between the computer and the user or external hardware devices such as keyboards and printers); (4) stability (ease of maintenance, upgrading and adaptation to new hardware platforms); and (5) usability (ease of use by the intended users). Software engineers seek to develop methods for improving the efficiency and reliability of programs in a cost-effective way.

The following subsections describe typical processes for software design.

[1] Requirements Analysis and Program Specification

Computer programs are functional[79] in that all software development processes begin

---

[77] In the *Pension* case (*T 931/95-3.5.1*, OJ EPO 10/2001 [441] Boards of Appeal of the EPO, decision of 8 September 2000), the court distinguished the invention from that of the *Sohei* (*T 769/92* (OJ 1995, 525)) on the ground that the method claim in the *Sohei* specified the functional features of the computer system. The CAFC held that the *Lowry* data structure claims were patentable because the data structure did sufficiently interact with memory to meet the functional relationship standard. See this study, 3.2 Software Patent in the US, and 4.4 BMPs in the EPO.

[78] See this study, 5.5 Copyright Protection for Software; Copyright Act of 1976 § 102(b), 17 U.S.C. § 102(b) (1993); www.law.berkeley.edu/institutes/bclt/pubs/ipnta/appenb.pdf.

[79] While the incremental nature of software innovation largely precludes patent protection, the functional

with a clear definition of the problem to be solved or the task to be automated.[80] This stage of the process identifies the goals of the users and the constraints of the hardware system. This stage is followed by the development of a user interface. After the software and an interface have been designed, programmers test its performance and reliability, and fine-tune the system. Many software products also require maintenance and updating to correct errors and enhance capability. They are subject to constant pressure for modification and additional features. For large collections of interconnected systems, constant changeability has to be dealt with.

In developing software to run at a law office, the designers must identify the various tasks that the computer program should handle, e.g. billing clients, filing documents, ordering supplies, preparing budgets, and filing court documents. The design team would interview the prospective users of the software to identify their needs. It is necessary to study the way that information flows and develop a schematic representation of the tasks to be programmed. This abstract representation of tasks will map data inputs and outputs and assess the hardware requirements.[81]

[2] Software Design

Processes for designing software systems have significantly changed over the past four decades as computer hardware has become more powerful and the problems to be solved have become more complex.[82] Most modern programs involve thousands or

---

nature of program precludes copyright protection. See this study, 6.1 [6] Subject-matter of the Protection.

[80] www.law.berkeley.edu/institutes/bclt/pubs/ipnta/appenb.pdf.

[81] In developing software to provide banking services through an automated teller machine (ATM), the design team will carefully arrange the necessary data, desired functionality, and constraints of hardware and security. They might develop tables or flow charts to understand the flow of information needed to accomplish the various transactions.

[82] The result of the software creation is a disk on which this instructions are saved. The creation process

millions of lines of computer code. Because of the enormous size,[83] writing an application program or an OS is a complicated process. They may be written by teams of programmers working together rather than by a single person. To facilitate collaborative work within teams, programmers sometimes develop a "flow chart" which shows the logical structure of the program. This represents what the program is supposed to do and how it will accomplish the intended tasks.

Because of the nature of software, small changes do not always have small effects. Computer programs and large collections of interconnected communication systems usually involve interactive and concurrent processes. Therefore, small problems can be intractable and the potential errors can be very subtle and hard to locate. Moreover, managing large-scale software projects, e.g. computer OSs, air traffic control, spacecraft simulation, robotics, etc, is a very difficult task. Programmers tried to divide complex tasks into component parts and solve each component's problems separately. This *procedure* oriented, functional decomposition, or top down methodology has served as a major paradigm for software design.[84]

Procedural programming combines returning sequences of statement into one single place. A procedure call is used to invoke the procedure. A program can be regarded as a sequence of procedure calls. A single program is divided into small pieces called procedures.[85] The main program passes data to the individual calls and the data is processed by the procedures. The flow of data can be illustrated as a hierarchical graph, a tree. With introducing procedures of procedures (subprocedures), program can

---

can be divided into a development phase and a production phase. In the production phase, the software is reproduced on diskettes or other devices. The main emphasis of the creation process exists on the development phase since a computer program can be copied without a loss of quality and with very little amount of material. www.law.berkeley.edu/institutes/bclt/pubs/ipnta/appenb.pdf.

[83] http://math.ucsd.edu/~fan/rep.pdf accessed 26 October 2001.

[84] www.law.berkeley.edu/institutes/bclt/pubs/ipnta/appenb.pdf.

[85] http://jeffsutherland.com/papers/Rans/OOlanguages.pdf.

be written more structured and error free.

- To enable procedures or group of procedures to be also used in other programs, they must be separately available.[86] *Modular programming* groups procedures into modules. Procedures of a common functionality are grouped together into separate modules. A program no longer consists of only one single part. It is now divided into several smaller parts which interact through procedure calls. Each module can have its own data. This allows each module to manage an internal state which is modified by calls to procedures of this module.

The programmer begins with a general description of the functions that a program is to perform. The programmer then outlines the program, specifying data structures and algorithms to be used. Such outlines are frequently expressed as flowcharts that show the relationship between the various modules or subroutines of the program. These modules are then separately further broken down so that the full logic of the program may be understood clearly.

As the complexity, scale, and need for updating of software projects have grown, the classical design methodology has become increasingly inefficient.[87] The many parts of very large software systems interact in many intricate ways. Classical top down designs often require significant redundancies. They are difficult to evolve and often require re-invention of many components. Software developers must typically build programs from scratch. Moreover, top down procedural techniques typically define data structures in one place. This can cause problems when the program is updated or revised. Whenever a data structure is revised, all subroutines or modules must be modified.

---

[86] Ibid.

[87] www.law.berkeley.edu/institutes/bclt/pubs/ipnta/appenb.pdf.

Object-oriented design

As a result of these limitations of classical programming methodologies, computer scientists developed the *object-oriented* paradigm.[88] The object-oriented paradigm enables programmers to design software by structuring relationships among independent *objects*, each of which represents a physical entity in the real world. Objects of the program interact by sending messages to each other. Objects are grouped within hierarchical structures. Higher-level classes *inherit* the attributes of sub-classes. Object-oriented systems can simplify complexity by encapsulating the internal data structures and procedures within objects.[89] The enclosure and protection of data is called encapsulation and results in the state of an object being hidden from procedures external to it. The data structures and procedures of particular objects can be changed without affecting other aspects of the large software system.

A programmer starting with an object-oriented program would not need to begin the software design process from scratch.[90] Such designs are more readily adaptable to changes in data structures and new variables than traditional top down designs. This adaptability of object-oriented software programs significantly reduces the risk of losing investments in software design when new features need to be added to a program. Moreover, object-oriented designs tend to produce smaller systems through the re-use of common mechanisms.

This re-use reduces the complexity of the software design and the cost of writing program code.[91] The re-use of computer code is valuable because it saves the cost of

---

[88] Ibid. The object-orientation emerged during the 1960s.

[89] http://jeffsutherland.com/papers/Rans/OOlanguages.pdf.

[90] There is an analogy. With a few minutes of orientation, an experienced pilot can safely fly a jet aircraft that he has never flown before. Once having recognized the properties common to all such aircraft, the pilot primarily needs to learn what properties are unique to that particular aircraft. Thus, if the pilot already knows how to fly an aircraft, it is very easy for him to fly a similar one.

[91] Mark A. Lemley and David W. O'Brien, *Encouraging Software Reuse*, Stanford Law Review, volume

rewriting program code and enables a particular software *object* to be refined, debugged and then re-used in various different programs.[92] It improves the safety and reliability of software, since programmers can use codes that have already been tested. The re-use of software *objects* across firms may enhance software compatibility.[93] Due to these advantages, object-oriented methodologies have increasingly become the norm in designing complex computer programs.

Algorithm and software design

The term 'algorithm' is used to mean a logic solution or a procedure. It can be expressed by numerical formula or by common language such as English or Korean. Since language, however, can be vague, more accurate methods of expression (e.g. logic sequential charts using specific logic symbols, flowcharts indicating the flow of processes) are generally used.[94] According to the *Proposal for a Directive of the European Parliament and of the Council on the Patentability of Computer-implemented inventions*,[95] issued by the European Commission, the term "algorithm" means 'any sequence of actions intended to accomplish a specific task' and the mere existence of an algorithm does not make an invention non-patentable.[96]

It is necessary to prepare algorithms for the processing requirements of particular information. This corresponds to creating a blueprint based on a technical concept for a new machine. Algorithms correspond to technical concept. When making the program,

---

49, No.2, January 1997 (hereinafter, Lemley, *Encouraging Software Reuse*).

92 www.law.berkeley.edu/institutes/bclt/pubs/ipnta/appenb.pdf.

93 Software re-use and compatibility will be discussed later in detail.

94 JPO, *Practical procedures.*

95 Commission of the European Communities, *Proposal for a Directive of the European Parliament and of the Council on the Patentability of Computer-implemented inventions*, Brussels, 20 February 2002, COM (2002) 92 final, 2002/0047 (COD), (afterwards, *Directive on the Patentability 2002/0047 (COD)*). Available at http://www.europa.eu.int/comm/internal_market/en/indprop/com02-92en.pdf.

96 See this study, 3.1 [2] [a] Mathematical Algorithm Exception.

most of the hard work is in analysing problems and inventing algorithms. Many kinds of specifications and instructions are created at this point. Therefore the devising of algorithms can be the most difficult process of software creation. After the algorithm has been devised, an appropriate program language may be selected.

[3] User Interface Design

As computers have become more versatile and available to a broader range of users, software development has increasingly focused on tailoring the program to the particular goals and the knowledge base of the intended users.[97] The design of computer program user interfaces exploits the field of computer-human interaction (CHI). CHI aims to understand how human beings process information so that products can better be designed to enhance usability. Recognizing that the study of human factors can aid computer system design, the computer industry has taken a strong interest in expanding this learning.

The technology of CHI has greatly changed the way that application programs are conceptualized and written. It has identified five human factor goals that programmers should seek to achieve in designing application programs: (1) minimize learning time, (2) maximize speed of performance, (3) minimize rate of user errors, (4) maximize user satisfaction, and (5) maximize users' retention of knowledge over time.[98]

---

97 www.law.berkeley.edu/institutes/bclt/pubs/ipnta/appenb.pdf.
98 Ibid.

[4] Generating Computer Code

After designing computer programs, software developers translate the design into binary code that is readable by computers. Before it is written directly into machine-level language, most programming is written in higher level languages such as FORTRAN, COBOL,[99] Pascal[100] and so on.[101] Computer programs written in these programming languages are often referred to as source code. These languages are easily understood by skilled programmers.

Source code was traditionally written by computer programmers. However, the process of writing source code has become automated. Computer programmers can now write code with the help of a computer, which translates higher-level design concepts directly into code. Source code writing with computer assistance may be faster and more efficient than writing it by hand. It also helps prevent programmers from making mistakes. Source code must be transformed into a machine-readable form to be executable by a computer. The transformation is accomplished by a compiler that translates source code into object code, the binary code.

---

[99] Common Business-Oriented Language.

[100] Pascal is a language developed by Niklaus Wirth of the Federal Institute of Technology, Zürich, Switz., in the late 1960s. It was intended to be a good educational tool for the systematic teaching of programming and to have fast, reliable compilers. Pascal strongly influenced many languages developed later, such as Ada. The language specifications of Pascal are concise, making it easier to learn than many other high-level languages. Complex data structures and algorithms can be described concisely by Pascal, and its programs are easy to read and to debug. 1996 Encyclopaedia Britannica, Inc.

[101] Special languages such as C++, Ada, Smalltalk, Object Pascal, and Java have been developed more recently to implement object-oriented designs. Ada is a high-level language whose development was initiated in 1975 by the US Department of Defence. Ada was intended to be a common language that could be used on the department's computers.

2.5 Characteristics of Modern Software Development

Innovation in software typically proceeds via a mix of new coding, modifications to some existing modules and/or subroutines, and re-use of others.[102] Patterns of improvements are greatly constrained by the need to preserve interoperability between program, system, and network components. The need for interoperability substantially constrains the range of options available to the second comer.

Microsoft has made both incremental and architectural changes to the Windows environment since the release of Windows 3.0 in 1991.[103] The original version used a 16-bit architecture, and Windows 95 and 98 use 32-bit architecture. The original Windows product was crude but has been enhanced significantly. With an installed base of 230 million users Microsoft has achieved critical mass with Windows. Windows has reached the point where any significant effort is not necessary to expand the market. 95% of desktop OSs are Microsoft Windows. While people complain about the need to reboot Windows frequently, new PC purchasers almost universally choose Windows as the default[104] OS because of the network effects offered in that environment.[105] The majority of general-purpose software and much special-purpose software are available for Windows. When the customers switch to another OS, they must pay the price of finding other application software, determining whether it interfaces with complementary hardware and software, and converting existing files to the new format.

---

[102] Julie E. Cohen & Mark A. Lemley, *Patent Scope and Innovation in the Software Industry*, California Law Review, Vol. 89:1, 2001 (afterwards, Cohen & Lemley, *Patent Scope*). This paper is available at www.law.georgetown.edu/faculty/jec/softwarepatentscope.pdf accessed 14 October 2001. This is also available at http://papers.ssrn.com accessed 10 February 2000.

[103] http://eplu.netgistics.com/PDF/1999-2000/Fall/BUSA541/cham1.pdf.

[104] 'Default' means what happens if any other choice is not made.

[105] With regard to network effect, see this study, 5.3. [3] The Nature of E-commerce, and 5.6. [2] Undesirable Aspects of Business Method Patenting. The value of a marketplace to its participants increases with the number of participants.

An OS derives most of its value from its capacity to function as a platform for other products. The higher the availability and sales of compatible applications, the higher the value of an OS.[106] The higher the sales of an OS, the more profitable are applications written for it. The demand for PC OSs is significantly influenced by network effects. The use of a common software platform creates direct network benefits such as easy file sharing, widespread familiarity with the same user interface and the compatibility of software applications across computers and computer users. Standardization of user interfaces promotes efficiency and reduces the need for undesirable retraining. An example of indirect network benefit is the greater incentive for software developers to invest in new applications as the number of platform users grows.

This section attempts to examine characteristics of software development, using OS as a primary example, since improvements in PC OS software have played and will play a central role in the development of computer industry.[107]

[1] Integration

A significant aspect of the evolution in commercial OS products is the continual integration of new features.[108] All commercially successful OS products aimed at the general computer user have expanded functionality over time. In a May 1998 antitrust complaint filed against Microsoft Corporation, the US Department of Justice contended

---

[106] Nicholas Economides, *Durable Goods Monopoly with Network Externalities with Application to the PC Operating Systems Market*, Quarterly Journal of Electronic Commerce, Vol. 1, no.3 (2000).
[107] Steven J. Davis, Jack MacCrisken, and Kevin M. Murphy, *The Evolution of the PC Operating System: An Economic Analysis of Software Design*, 29 June 1999, available at http://www.nber.org/papers.
[108] Examples of the integration are the graphical user interface, disk management and data compression utilities, memory management utilities, fax and email utilities, support for LANs, integrated audio support and web browsing functions. Davis, *Economic Perspectives*, p. 7.

that the integration of Internet Explorer into Microsoft's Windows OS hindered competition with other Internet browsing software and unfairly helped preserve Microsoft's dominance of the OS software.[109]

There are three basic forces behind the integration in the evolution of commercial OS products: (1) the need to keep pace with advances in computer technology, (2) the need to simplify computer use, and (3) the desire to stimulate new and improved applications for the OS.

Integration by keeping pace with technological advances

Rapid innovation is the characteristic of computer technology. The resulting changes cause frequent upgrades in complementary products, especially OSs. By the early 1990s, there had been made great advances in speed and functionality of PCs. Advances in the efficiency and miniaturization of chips, batteries, hard drives and other components had been made.

In the mid-1990s, the Internet greatly influenced the computing environment. The Internet created a surge in the demand for PCs and servers. This development has generated incentives to add new capabilities and features to OS products. For example, Microsoft made PC-DOS 1.0 for the original IBM PC in 1981. When IBM added a hard drive to the PC a year later, version 2.0 was provided to support the additional storage medium. DOS 3.1 adapted the PC for use on LANs. DOS 3.2 and DOS 3.3 supported the new 3-1/2 inch, 720-kilobyte and 1.44-megabyte floppy disks respectively. And when the 386 microprocessor arrived, DOS version 3.3 was supplied to support it.

The ability to manage huge amounts of data and display complex graphics has

---

[109] Steven J. Davis, *The Evolution of the PC Operating System.*

decentralized publishing from the factory to the desktop. Rapid number crunching[110] and super-cheap memory have transformed the spreadsheet into an all-purpose tool for business, finance and science. The Internet provides PCs with the access to a huge amount of data. New demands on OS products to manage more hardware and more software with more complicated user interfaces have come with the expansion of functionality.[111]

Integration to simplify computer use

A second key force behind OS integration is the need to make computers easier to use. Computer OSs have evolved to facilitate computer use. Most customers greatly value the ease of use even for the products that depend on very complex technology. They value software for performance, ease of use and compatibility with other elements of a computer system.[112] They also expect quick solutions when problems happen. As the market for PCs has expanded, the market pressure to make them easier to use has intensified. Much of the growth in demand now comes from untutored customers. This changing market reality has expanded the PC OS to include elements such as software to run peripherals, e.g. scanners and DVD players. Integration often involves distinct components that could be sold separately.[113]

Integration facilitates the development and introduction of additional non-integrated products.[114] It simplifies end-user experience by helping to manage the interacting components in a system product. Integration also lowers customer support

---

[110] Number crunching is the process of calculating numbers, especially when a large amount of data is involved and the data is processed in a short space of time.
[111] Davis, *Economic Perspectives*, p. 17.
[112] Ibid., p. 29. They are not interested in the internal design of software, i.e. *how* it accomplishes tasks or achieves ease of use and compatibility.
[113] Ibid., p. 40.
[114] Ibid., p. 45.

costs. For many software companies, customer support is a major cost.[115] This reality compelled Microsoft to emphasize design advances that would reduce customer support costs. The costs of customer support can be greatly reduced by careful integration of software functionality into the OS. OS integration also satisfies the need of customers who prefer to deal with a single vendor, especially when they are managing many interacting components in a complex system. In a complicated system, when problems happen, customers can easily find the solution if a single vendor supplies all the components.

Due to the explosive use of the Internet, web browsers are now routinely integrated into OS products. Internet Explorer, which is the web browsing technology developed by Microsoft, is integrated into Windows 98 and later versions of Windows. IBM as well incorporated its own browser into OS/2.

Integration to attract software developers

A third key force behind integration is the desire to encourage software developers to create new applications for the OS and thereby enhance the OS's value.[116] Microsoft's Word 97 uses Internet Explorer to convert automatically any typed web address (URL) into a live link. Clicking on the address opens the web page in the browser (Internet Explorer), if the user is connected to the Internet.

Integration facilitates the development of new applications for the system product. The integration of key software building blocks (APIs) into the PC OS promotes innovation by reducing innovation costs. Most developers of application software leave

---

[115] Ibid., p. 50. As of the early 1990s, Microsoft supported about 60,000 customer inquiries per day, including 20,000 phone calls that should be handled by product engineers. Microsoft personnel in the customer support division actually outnumbered the firm's software developers. As of 1993, customer support costs of Microsoft equalled 20% of the gross revenues from Windows.

[116] Ibid., p. 19.

routine tasks (e.g. file management, memory management, graphical displays, and video and/or audio management) to the OS. This relieves developers from the need to re-invent the wheel in each application and enables them to focus on their fields of expertise. Thus, OS integration simplifies the use of interacting components and reduces innovation cost of applications, resulting in facilitating the introduction of new PC applications and the standardization of software development.[117] Integration also facilitates on-line distribution and reduces the cost of distributing applications products.

Integration becomes more useful when the use of the stand-alone elements increases and the stand-alone elements interact with other elements in a complicated way.

[2] Software Design Flexibility

Software modules are connected together into unified programs.[118] Each software product is built up from low-level modules. Modules at each level are called by higher-level modules until the desired functionality of the end product is achieved. In this manner, all software is built up layer by layer.[119]

As a result of this layering, software has an inherently malleable and modular structure that gives software developers wide freedom in combining different functions into software products. Because of this malleability, the same functionality can be achieved in many ways.

Modularity permits an efficient exploitation of the division of labour by allowing

---

[117] Ibid., pp. 49, 52.
[118] Ibid., pp. 22-25.
[119] Computer programs are compilations of sub-components, compilations of behavioural components, compilations of useful behaviours. See this study, 5.2 Characteristics of Software.

the different tasks of a software project to be allocated to the most efficient producer. Modules can be assigned on the basis of expertise.[120]

[3] Componentized Design Architectures

Componentization refers to a modular design architecture[121] that structures the interactions among the elements of a software system.[122] Componentization can facilitate product development, design and testing. This results in the reduced cost of software products. Componentization can also facilitate successive improvements in a large and complex system. This design architecture prescribes the pathways along which components communicate one another.

A few analogies can clarify the abstract concept involved in a componentized design. We can consider two alternative design architectures for an integrated TV-VCR system. One is a closed system housed in a one-piece construction. This architecture enables the machine to be compact, easy to use and cheap to manufacture. A second is a modular construction with separate units for the TV and the VCR. These units are only integrated in the sense that they are connected and work together. In the second design it is easy to upgrade individual pieces of the system. If the modular system is open, it can accommodate new components such as a DVD player later on. Thus, the modular

---

[120] Bruce Kogut and Anca Meitu, *The Emergence of E-Innovation: Insights from Open Source Software Development*, http://jonescenter.wharton.upenn.edu/events/software.pdf accessed 10 November 2001.

[121] The architecture of a system is often defined as the set of subsystems and their mutual relations. Ommering defines 'architecture' as 'everything a single person or group of people need(s) to do to let a large team develop a product or family of products successfully'. See www.win.tue.nl/xootic/magazine/mar-2000/vanommering.pdf accessed 14 October 2001. Rob van Ommering, *A Composable Software Architecture for Consumer Electronics Products*, Xootic Magazine, March 2000; Software Engineering Institute, Carnegie Mellon, *How do you define software architecture?* http://www.sei.cmu.edu/architecture/definitions.html; Davis, *Economic Perspectives*, p. 30.

[122] Computer programs are compilations of sub-components and compilations of behavioural components. See this study, 5.2 [4] Programs are Industrial Compilations.

design makes a system more flexible, even though it may be more costly in the short run.

We can find another analogy in a fighter jet. That is a complicated technological system with many subsystems and interacting components. Each component must properly work and interact in order for the whole system to accomplish the required capability. Intelligent design architecture will organize many small teams which focus on a subsystem or component. This approach enables development to proceed in many fronts solving many small problems simultaneously.

A componentized design has many benefits. First, componentization facilitates code sharing across same-generation programs and code re-use in new products. Code sharing has the following advantages:

- It reduces the need to reinvent for each program and lowers development costs.
- It reduces product testing and debugging costs by simplifying the interactions among blocks of software code.
- Developers can focus their effort on optimizing a component's technical performance.
- Code sharing across products (e.g. Microsoft's Word, Excel and PowerPoint) helps to harmonize the user interface and the user experience.

Second, componentization enables it easier to integrate new functionality into existing software. When the new functionality causes problems, a componentized design easily identifies the source of the problem. This advantage becomes more important when software products and systems become larger. Intelligently componentized software is more flexible to integrate a new functionality into an existing software system.

Third, componentization makes it easier to maintain the backward compatibility of

platforms as they evolve. Computer users can continue to use a redesigned software component, if the component's interface remains unchanged. This advantage is especially valuable in a product like Windows that serves as a platform for thousands of application software. In order to attract users, a new version Windows must continue to serve as a platform for the existing stock of applications.

Fourth, componentization facilitates a small-team approach to software development since it enables to break a project into manageable tasks.[123] This benefit is obviously valuable in the development of large-scale software products or integrated collections of application software. Microsoft platform products (e.g. Windows) and business applications (e.g. Microsoft Office Suite) are among the largest software systems offered by any mass-market software vendor. Since componentized design architecture facilitates a small-team approach, it is especially valuable in the development and improvement of these large software systems. Componentized development process, thus, enables large teams to work like small teams.

Since Netscape Navigator was not componentized by early 1997, the componentized nature of Microsoft's Internet Explorer technologies in Windows 95 version became an important factor in Intuit's (Intuit is the maker of Quicken financial software.) decision to switch from Navigator to Internet Explorer as its primary browser. It is clear that the componentized design of Internet Explorer made it more attractive to some providers of complementary application software and Internet services. The value of Windows as a software platform was enhanced in this process due to the componentized nature of Internet Explorer.

Microsoft has emphasized componentization over the past decade in line with the

---

[123] Davis, *Economic Perspectives*, pp. 36-38.

increasing scale and complexity of products such as Windows and Office.[124] The company has sought to benefit from code re-use, to optimize the inner workings of a software component and to harmonize features and performance across software products. Since the late 1980, Microsoft has made a conscious effort to harmonize the user interface, feature and performance characteristics across its major software products. This effort focused on user interface, code sharing and functional integration. This trend is especially obvious in the MS Office Suite. Originally, Word, Excel and PowerPoint were independent products with little shared code and limited integration. After discussion in Cusumano and Selby (1995), much had changed by the mid-1990s. The Office products now share much of their code and are closely integrated. Their current development is now closely coordinated. The move toward componentization at Microsoft is a long-term effort toward harmonization of features and functional integration across software.

However, designing intelligently componentized software is difficult, time consuming and expensive.[125] It is especially difficult to componentize an existing large software product that was not originally designed that way. An ill-conceived componentization may increase the number of calls between interacting components, resulting in slow execution. Therefore, software components should work together in a way that avoids excessive demands on the overall system.

[4] Software Re-use

Software re-use involves integrating software components from existing software

---

[124] Ibid., p. 39.
[125] Ibid., p. 33.

systems into the development of new software.[126] Such components include code, system architecture, documentation, user interfaces and data. There are two types of re-use: opportunistic re-use and systematic re-use. Opportunistic re-use is the informal practice of exploiting code from existing systems and modifying it to serve the design goals of a new program. Individuals and small software companies routinely practice opportunistic re-use. Systematic re-use of software components creates new programs from software designed to be incorporated and re-used within other programs. Systematic re-use can be internal or external. Large companies such as AT&T, IBM, HP and Microsoft already have internal re-use programs. External re-use is uncommon even in large corporations.

Because of its great benefits, software re-use is increasing. As the complexity and scale of software grow, software development becomes more difficult and the importance of re-use becomes greater. The US government undertook an ambitious program to create large national software asset repositories,[127] and in 1994 the National Institute of Standards and Technology (NIST) announced a $150 million program to develop a nationwide external market for tradable software components.

Re-use benefits

Systematic software re-use can improve the quality of components, increase the productivity of the developers, and reduce the time to market of the developed product. The quality of code is improved over time as bugs in a program are identified and corrected. Re-use facilitates this quality improvement process, because each code is used over the life span of several applications, and because software companies have a

---

[126] Lemley, *Encouraging Software Reuse*.

[127] E.g. the Asset Source for Software-Engineering Technology ("ASSET"), which is available on the Internet and provides re-usable software components. See http://source.asset.com/asset.html.

greater interest in catching errors in re-usable components than in software that is used once and thrown away.

Re-use costs

The creation and maintenance of re-usable software components is more costly than the creation of ordinary computer programs. These additional costs come from the efforts to create an infrastructure for re-use, purchasing re-usable components, establishing libraries [128] and search techniques, maintaining and upgrading components, and educating programmers in the practice of re-use. The most important costs are the costs of creating re-usable software components and of integrating re-used components into new products.

However, the cost of integrating the re-usable software component with other software is considerably lower than the cost of developing and integrating a comparable new component from scratch.[129] Therefore, although it costs more to create re-usable components initially, overall cost reduction by re-usable components outweighs the initial cost. In other words, while the initial development of modular re-usable software is more expensive than the development of code from scratch, the savings from re-use of the software outweighs the additional costs.

[5] Software Bundling

In order to meet the demand for complementary applications, firms bundle multiple

---

[128] With regard to electronic repositories of re-usable software components that already exist, see this study, 6.1 [5] [c] Registration and an Automatic Licensing System.
[129] Lemley, *Encouraging Software Reuse*, p. 266.

software features into a single package.[130] Multiple features or applications are packaged together or distributed jointly. Software vendors often bundle large collections of utilities and features with OS and platform products. A large collection of distinct software features is offered on single CD-ROM. In 1993, Microsoft Windows included disk compression features and fax functions.

The bundling as well as integration of new features and functions into OS products spurs innovation in the computer industry. By making PC systems easier to set up and use, integration facilitates the introduction of new products.[131] Bundling applications and utilities with OS products also leads to wider and cheaper distribution of software. It increases consumers' benefit by stimulating the development of application software that would otherwise be unprofitable. In sum, integration and bundling of new features and functions into PC OS products are highly beneficial for consumers and a major stimulus to the growth and innovation in the computer industry.

2.6 Development of Software Market

In order to find a regime which is in harmony with the software market, it would be essential to understand the characteristics of modern software market.

In 1969, IBM ended software bundling. Before this, software was bundled with hardware and its price was not considered separately by consumers. In these early stages, software could not be bought or sold independently of the hardware. When IBM stopped software bundling with hardware, other companies could enter the market and the software industry was born.

---

[130] Davis, *Economic Perspectives*, p. 77.

[131] Integration of widely used features and software development tools into OS products promotes a standard setting as well.

[1] Standardization

We live in a world built on product standards, where almost everyone is using the standard technology, and is making minute and incremental changes.[132] We can exchange e-mail because of a complex web of standardized Internet protocols. A great deal of the information economy is driven by standards. There are two types in the development process in relation to software. One is for software to be sold to only one customer, and the other for software to be sold to many customers.[133]

The development of individual software is characterized as individual, customer-oriented and single-unit production. The software is usually developed for a single project and can be re-used to a limited extent for other projects. Individual customers influence the development process. The development cost of this kind of software is high because the software is mostly sold only once and the fixed costs are not shared among customers.

Computer programs are also developed to sell to a mass market. Potential customers do not influence the development of the software. A developer can produce standard software. The development costs are lower than in the case of individual software because the costs are shared among multiple customers. Because of the high development costs of individual software and the advantages of standard software, the market share of individual software is diminishing and that of standard software is increasing. With this trend; the importance of adaptation of standardized software is also

---

[132] Edited by Rochelle Dreyfuss, Diane L Zimmerman and Harry First, *Expanding the Boundaries of Intellectual Property*, Jerome H. Reichman, 2001, Oxford University Press. (Afterwards, Dreyfuss, *Expanding*), p. 81; http://www-2.cs.cmu.edu/~amulet/papers/uihistory.tr.html accessed 10 October 2001. .

[133] Jurgen Bitzer, *The computer software Industry in East and West: Do Eastern European Countries Need a Specific Science and Technology Policy?* http://www.diw-berlin.de accessed 13 October 2001.

rising. In the case of standard software, further development and repair of faults are possible.

With the growing number of users who can solve a problem with the same software, the degree of standardization increases. The standardization degree is influenced by the complexity of the application field, international standardization and the strategies of the software companies. Standardization degree of software that offers solutions for daily communication is very high (e.g. word-processing software) and that of individual software is very low. Software products with a high degree of standardization are mostly marketed worldwide whereas products with a low degree of standardization are mostly sold nationally. With a rising degree of standardization of software products, the possibility of selling them worldwide grows.

[2] Standardization and Innovation Trade-offs

Innovation has been influenced by standardization. Much of the innovation taking place now in the telecommunications and computer areas is standards-based.[134] There have been conflicting economic goals. The more regulation imposes integration and compatibility, the more it stifles diversity and innovation. The need to abide by a standard imposes limits on firms' product design choices.[135] Limits on design choices can lead to static losses from the reduction in variety. These can also lead to dynamic losses since firms are prevented from certain paths of research and development (R&D) that could result in innovative new products that could not comply with the standards. These limits impose costs both when a new product is created and when a new

---

[134] Adams Jaffe, Joshua Lerner, and Scott Stern, eds., *Innovation Policy and the Economy*, Volume I, MIT Press, 2001. This paper is available at http://haas.berkeley.edu/~shapiro/thicket.pdf.
[135] Dreyfuss, *Expanding*, p. 88.

generation having greatly enhanced capability is introduced. Once a particular program became established, competitors would have little incentive to make innovative improvements, because the potential market could be significantly reduced by the large number of users locked in to the established system. Without competition, there would be less incentive even for the original programmer to make improvements.[136] When products are compatible, however, a consumer does not fear being stranded when he purchases a particular product. When a consumer buys a computer, he knows that it is compatible with general programs.

Conversely, the more the market promotes diversity and innovation, the greater the risks of incompatibility and disintegration are.[137] Without standards, consumers are confused and they become reluctant to buy.[138] When a computer buyer is considering changing to a different CPU and OS, he must make a decision to choose between the expected benefits from switching to new one (the innovation) and abandoning investments in old one (the standard).[139] The investments include not only the cost of the current system, but also the time spent in learning how to use it and the cost of replacing other programs already purchased. PC/Windows users have invested massive amounts of time, labour and other resources in a very popular and convenient computing environment. Any company trying to persuade consumers to buy a different product must offer advantages strong enough to induce users to abandon those investments.

---

[136] Dennis S. Karjala, *Copyright, Computer Software, and the New Protectionism*, 28 Jurimetrics Journal 33 (Autumn 1987).

[137] http://brie.berkeley.edu/~briewww/pubs/wp/wp79.pdf accessed 13 October 2001. *Islands in the Bit-Stream: Charting the NII Interoperability Debate*, Francois Bar, Michael Borrus and Richard Steinberg, Berkeley Roundtable on the International Economy (BRIE), University of California, Berkeley, BRIE Working Paper 79, 1995.

[138] Dreyfuss, *Expanding*, p. 88.

[139] Barry Fagin, *Standardization/Innovation Tradeoffs in Computing: Implications for High-Tech Antitrust Policy*, www.faginfamily.net/barry/Papers/tradeoffs.htm accessed 15 October 2001.

In the market for highly standardized application software, network effects play an important role. If more people use the same software, the transfer of data becomes easier and the benefits for every user become greater. When all users are on a single network, the size of the network is maximized and network benefits are realized.[140] In communication networks, users benefit from the situation that every user can communicate with one another. Users benefit from the fact that firms supplying components have access to a large market for their software. This may lead to increased entry and variety, and greater price and innovation competition in the supply of individual components.

A network provides the pathways for interaction among different users or terminals. Standards establish a common mode of interaction, such as use of the English language, which enables users to understand each other's communication.[141] Networks and standards are interrelated in the sense that every network is based on certain standards. Both have the distinctive characteristic that their value tends to rise as more users participate. With the development of communications and computer industries, the importance of standards and networks has increased greatly.

Networks as well as standards impose costs on the consumer who switches to alternative providers. Once consumers purchase a primary good such as PC hardware or an OS, they often invest heavily in complementary products, such as peripherals[142] and applications software. They may also develop expertise and a stock of files. The current users of such products are referred to as a system producer's "installed customer base". Unless competing systems are compatible with the installed base's peripherals as well

---

[140] Dreyfuss, *Expanding*, p. 88.

[141] US Federal Trade Commission Staff, *Anticipating the 21st Century - Competition Policy in the New High-Tech, Global Marketplace*, Volume I, May 1996, (afterwards, US FTC, *Anticipating the 21st Century*), Chapter 9. Networks and Standards.

[142] Peripherals include external disk drives and printers.

as applications software, expertise, or files, the installed customer base may be locked in to the incumbent system because switching to competing system entails the cost of replacing the complementary assets. Compatibility reduces consumers' costs of switching to rival products and thus facilitates entry and competition in the market.

Internet standards development

As noted above, the Internet is based on the original protocols of ARPANET.[143] In spite of problems[144] discovered with these standards, because of the enormous amount of resources invested in existing Internet standards, the development of a new set of protocols from scratch is not worth considering. The next set of Internet protocols (IPv6) is compatible and interoperable with the current one (IPv4). Extensive attention has been given to the transition mechanisms and backward compatibility. Introduction of an incompatible standard has not been considered due to the tremendous costs it would impose on society.

Programming language design and adoption

Early computing languages such as FORTRAN and COBOL were such significant improvements over existing alternatives that they became widespread very quickly. Research into programming languages, however, found their shortcomings. These include lack of expressive power and unnecessarily difficult syntax.[145] Advances in software engineering introduced powerful concepts e.g. data abstraction, encapsulation and object-oriented design. Succeeding versions of FORTRAN have been released in

---

[143] www.faginfamily.net/barry/Papers/tradeoffs.htm.

[144] The inefficiencies of a class-based address space are now a cause for significant concern, and the present pool of 32-bit addresses can be depleted within the next several years. Security, authentication and privacy become extremely important and require support from the network itself.

[145] 'Syntax' is the rules that state how words and phrases must be used in a computer language.

line with the advances in programming language design and software engineering. Nonetheless, very few of the proposed programming languages have been used on a large scale to develop software, and still fewer have been proven successful, despite their technical merits. This is because of the enormous resource loss that programming language change requires.

Windows vs. anything else

Microsoft has concentrated its effort on building market share, resulting in significant compatibility and convenience advantages to consumers.[146] Switching to a new OS, however, would impose huge costs on consumers. Companies who wish to persuade consumers to adopt a new model of system must offer extraordinary performance advantages, low cost and minimal compatibility problems.

The Linux Operating System

Linux is an UNIX-like OS.[147] Linux is a complete OS, which includes a graphical user interface, an X Windows System, TCP/IP, the Emacs editor, and other components usually found in a comprehensive UNIX system. It is available for free on the Internet. What is remarkable is that it is possible for users to access the complete source code and to find a programmer on the Internet who can help them when they meet problems. Due to this nature, thousands of users all over the world have contributed to the development of Linux so far.[148]

---

[146] www.faginfamily.net/barry/Papers/tradeoffs.htm.

[147] http://whatis.techtarget.com/definition/0,289893,sid9_gci212482,00.html and http://www.schwarz-online.com/chris/computer.html accessed 8 January 2002.

[148] Torvalds realized that his programming capability alone would not be enough to complete his OS. He decided to release his program to the Internet community. This made Linux be developed far beyond a personal project.

Linux's kernel[149] was first written in 1991 by Linus Torvalds.[150] Even though it was only a personal project, he had accomplished a working kernel. (The kernel is the core component of a Unix operating system.) To complete the OS, Torvalds used the system components developed by members of the Free Software Foundation for the GNU project. Torvalds created a simple scheduling algorithm that enabled two processes to run simultaneously, one printing 'A' repeatedly and the other printing 'B' repeatedly.[151] Although this may not appear to be significant, the scheduling algorithm is one of the key elements of a multitasking OS. Torvalds added more features on this basis.

Since Linux is distributed using the Free Software Foundation's copyleft stipulations, any modified version redistributed must in turn be freely available. Unlike Windows and other proprietary systems, Linux is publicly open and extendible by contributors. Developers can write programs that can be applied to other OSs. Linux comes in versions for all the major microprocessor platforms including the Intel, PowerPC, Sparc, and Alpha platforms. Thus, it is often suggested that Linux is a potential publicly-developed alternative to the desktop predominance of Microsoft Windows.[152]

Because the entire source code was available for free to anyone who wants to use the system, the work drew an immediate attention of curious programmers when Torvalds announced it. Releasing the source code for free led other enthusiastic

---

[149] The kernel is the central part of the OS.

[150] He was then a 21-year-old computer science student at the University of Helsinki in Finland. He wanted to write an Unix-like system for his own PC. Linux grew out of Linus Torvalds' dissatisfaction with the OSs of the Intel 386 processor. See http://firstmonday.org/issues/issue5_3/kuwabara/index.html and http://www.developer.ibm.com/library/articles/schenk1.html *Linux: Its history and current distributions* accessed 8 January 2002.

[151] See http://www.developer.ibm.com/library/articles/schenk1.html.

[152] However, it remains far behind Windows in numbers of users, even though Linux is used by users who are familiar with UNIX. http://whatis.techtarget.com/definition/0,289893,sid9_gci212482,00.html.

programmers to provide Trovalds help, support and feedback. Thus, even though the first official version was released in 1994, changes were being made on a daily and weekly basis and Linux continued to become a powerful OS.

As open-source software, Linux is remarkable in that its kernel alone consists of nearly one million lines of code. The size of the Linux project is unprecedented in the history of software development. Thousands of programmers have voluntarily participated at the daily development of numerous components and functions that are involved in the OS. There has never existed centralized organization to communicate between Torvalds and the thousands of contributors. Without prescribed tasks and responsibilities, each person decided what to do at the moment. Nonetheless, the Linux project has proceeded at a remarkable speed with updates and version releases on a daily and weekly basis at times. Although Linux was conceived five years after Microsoft began in the development of Windows NT, Linux is considered a competitive alternative to NT. Linux is said to have surpassed Microsoft Windows in many respects of performance as well as reliability. It is estimated as being installed at more than 3 million users worldwide.[153]

Linux is a hierarchical system which consists of at least two distinct levels of local interaction. One level is the source code. Linux is composed of units and subunits. The units of code form an immense web of interdependent functions.[154] This connectivity enables the OS to process a wide range of input from the user, and to respond with logical behaviours in a remarkably interactive way.

---

[153] http://firstmonday.org/issues/issue5_3/kuwabara/index.html accessed 8 January 2002. *Linux: A Bazaar at the Edge of Chaos.*
[154] Ibid.

[3] Compatibility and Interoperability

The need for compatibility creates an imperative for various firms to work together to develop, establish and promote standards.[155] Sometimes this cooperation is strictly among companies selling complementary components that work together to form a system.[156] Companies that compete directly with one another often agree on compatibility or standards to build sufficient support for a new technology.[157] Even enemies team up in the software industry to promote new standards.[158]

Software has innate characteristics that encourage users to remain with the same vendor. One important consideration is *backward compatibility* that is a compatibility with the learned skills associated with previous versions of a program, and a compatibility with file formats that were created with previous versions of the program.[159] If a new spreadsheet or word-processor is not compatible with the users' old files, the users are unlikely to change. This is why most software products provide some degree of *backward compatibility*. For enterprises that invest much money in changing computer systems, the software compatibility between the software they use now and the new software is very important because they want to continue to use their accumulated data. A lack of compatibility causes high costs for the transformation of data into the new format required by the new software. Customers are, therefore, forced

---

[155] Dreyfuss, *Expanding*, p. 82.
[156] Intel and Microsoft teamed up to make sure that their chips and OS function in harmony. See Dreyfuss, *Expanding*.
[157] For example, Sony and Philips jointly established and licensed the CD standard. See Dreyfuss, *Expanding*.
[158] In 1997, Microsoft and Netscape, which are hardly known as friendly partners, agreed to include compatible versions of Virtual Reality Modeling Language (developed by Silicon Graphics) in their browsers. See Dreyfuss, *Expanding*.
[159] Stan J. Liebowitz and Stephen E. Margolis, *Causes and Consequences of Market Leadership in Application Software*, A Paper presented at the conference: Competition and Innovation in the PC Industry, 24 April 1999.

to buy software from their former supplier because of the compatibility. These customers are called "locked-in-users". However, compatibility is obviously desirable for users, because it allows them to use application programs interchangeably among computers and makes a larger number of application programs available. With the growing number of users, the price of the software may be lowered because the development costs can be distributed among the users, and thus the degree of standardization is increasing. The problem of compatibility is to what extent the interface software can be copied in order to achieve compatibility.[160] If it were not for compatibility, the dominant player in the market would have a monopoly not only on his software but also on all software developed by third parties to run on his computer or system. The important issue is whether compatibility can be achieved in a socially optimal way, without infringing the protection given to the first comer's innovative software. Most commentators agree that copying of another OS should be permitted to the extent that is necessary for compatibility.

On the other hand, interoperability is the ability of different interconnected 'systems'[161] to work together in a predictable and coordinated fashion to accomplish a common purpose.[162] Compared with simple interconnection,[163] interoperability presupposes a higher level of logical compatibility necessary for two systems to work in harmony. The more complete the compatibility, the greater the interoperability. To enhance computer interoperability, emphasis should be focused on how different machines can run the same software, and how different programs can exchange files and

---

[160] See Dennis S. Karjala, *Copyright, Computer Software, and the New Protectionism*, 28 Jurimetrics Journal 33, Autumn 1987.

[161] The 'system' includes components, sub-systems, software, databases, etc.

[162] http://brie.berkeley.edu/~briewww/pubs/wp/wp79.pdf.

[163] While interconnection comes from the telecommunications world, interoperability comes from the computer world. See 'Bar, Interoperability Debate'.

work together.[164] In the software market, competitors try to benefit from network effect.[165] Firms developing OSs have strong incentives to be compatible with existing applications, and to promote the creation of applications compatible with their OS. It is common to release hundreds of new applications simultaneously with the release of an OS in order to raise the network effect of the OS.

There are three types of compatibility: (1) full compatibility between the OS and its applications, (2) compatibility of the new OS with existing applications (*backward compatibility*), and (3) compatibility with an old OS with applications written primarily for a new OS (*forward compatibility*).

When a new OS is introduced, the OS is expected to be compatible with existing applications. This is called *backward compatibility*. The new OS benefits from being backwardly compatible. This may force the new product to be more similar to the old one.[166]

[4] Network Externalities

The software market has the nature of strong *network externalities*, because the total social value of a software program increases as more consumers use it.[167] There are three types of network externalities: *interconnection* externalities, *interoperability* externalities, and *convenience* externalities. The typical example of interconnection externalities is the telephone network and the Internet, where the product is the

---

[164] The established base of trained software users reinforces the need for interoperability. See this study, 5.3 [1] The Nature of the Software Industry.
[165] Nicholas Economides, *Durable Goods Monopoly with Network Externalities with Application to the PC Operating Systems Market*, Quarterly Journal of Electronic Commerce, Vol. 1, No.3 (2000).
[166] Firms should balance the benefits of extensive network externalities created by backward compatibility with the potential disadvantage of more intensified competition.
[167] Lemley, *Encouraging Software Reuse*, p.287.

connection between customers of the network.

• Interoperability externalities emerge where a product's value is a function of its compatibility with other products in the market. For example, an electric power plug is of no use if it cannot connect firmly with most wall outlets. The design of electric plugs, thus, is substantially limited by the design of wall outlets and vice versa. Convenience externalities exist where a product becomes more desirable as more consumers learn to use it. This is because consumers become less willing to learn how to use new products.[168]

In general, software shows strong interoperability characteristics. This is particularly true for OSs. In the market for OSs, standardization is inevitable. Even though a competing OS is technically better than the standard, users will not purchase the system if it is not compatible with a wide variety of application programs. Thus, once a program becomes the standard, it remains in that position for a long time. Even in the applications market, where network externalities are not so distinct, interoperability with OS or with competing programs affects consumer product selection.

There are many ways to meet consumer demand for interoperability. Competitors can agree to establish a standard of compatibility among their product. Governments can require a standard. Joint standard setting organizations are relatively common in the computer industry. Individual companies can sell or give away *open platform* systems to encourage competitors to write programs that run on this platform.

---

[168] For example, the typewriting industry has used the QWERTY keyboard design for over a century even though it is arguably inferior to competing designs because training typists on a new keyboard would be too difficult and costly.

[5] Standard and Individual Software Market

The standard software market

The market for highly standardized software is dominated by a small number of enterprises, e.g. Microsoft for office application software.[169] The position of the market leader in such segments is very strong due to the reasons described above. Because of the high development cost, only large enterprises exist on this market. The market segment for highly standardized application software is protected by high entry barriers. In order to enter the market, a competitor should be able to offer at least the same quality of products as those of current companies, a lower price and a good reputation. Even equipped with these requirements, it would take some years to capture a market share. Thus, large financial resources are needed for a successful market entry.

In the market with a lower standardization degree, the intensity of competition drops. It is easier to enter the market and to establish a business. The number of SMEs increases because of their proximity to their customers.

In the system software market, hardware producers play a dominant role because they have the knowledge of the hardware and they have developed the system software, e.g. OSs.[170] Only Microsoft could successfully introduce an OS[171] without being a hardware producer. It is believed that this was possible due to the support of IBM in the early years. Because of the special knowledge and high quality required, the number of suppliers in the market for system software is small.

---

[169] http://www.diw-berlin.de.

[170] System software such as OSs, programming tools, and security utilities links the hardware with users and makes a computer usuable. System software controls the teamwork of the hardware components. In a PC the system software controls the CPU, memory, floppy disk, graphic device, hard disk and a printer. The software can control the teamwork of multiple computers and a number of industrial robots in a production process.

[171] This OS for the PCs of IBM was named as MS-DOS.

The individual software market

Standardization in the market for individual software is difficult due to a rising specificity of the requirements of users. Because of high development costs, enterprises choose this kind of software only when standard software cannot provide appropriate solutions.

2.7 Summary and Implications

Major innovations in the development of computer programs occurred in the 1950s, 1960s, 1970s. Since 1980s, software development has mainly been characterized by the trend of integration, bundling, componentization and re-use.

Continual integration of new features to commercial OSs has been made to keep pace with advances in computer technology, to simplify computer use, and to encourage the development of improved applications. Thus, OS integration simplified the use of interacting components and reduced innovation cost of applications. This facilitated the standardization of software development.

In the development of large software systems, componentized design architecture has developed by the needs to integrate new functionality, to maintain compatibility, and to facilitate code sharing and a small-team approach to software development. Componentization gave more flexibility in software design and facilitated software re-use. Innovations in componentization and software re-use generally exist in the combination of elements that are new or well-known.

The need of compatibility and interoperability has encouraged standardization:

The degree of standardization in software market has increased with the growing number of users who can solve a problem with the same software. The standardization trend has been accelerated by the emergence of the Internet. The growing need to communicate with others through the Internet has brought high degree of standardization. The need to abide by a standard, however, imposes limits on firms' design choices. Innovations are now substantially constrained by the need to preserve compatibility between programs, systems and networks. Because of the maturity that modern software industry has achieved, many improvements of software are increasingly difficult and expensive.[172]

Therefore, due to the trend to integration and componentization as well as the need of compatibility, innovations in software development can be characterized as sequential, cumulative and incremental.[173] It is the routine engineer's cumulative and sequential working out of common technical trajectories that increasingly drives the software development.[174] Thus, the development of software is mainly made by small, grain-sized and sub-patentable innovations.[175]

On the other hand, because in the context of today's software production it is basically difficult to keep innovative industrial designs secret from potential competitors once embodied in products distributed in the open market, copiers can reduce their natural lead-time to zero without incurring any significant R&D costs.[176] Moreover, the development of the Internet has substantially reduced the cost of copying, shrunk lead-time, and increased the risk that small scale innovators will keep their know-how

---

[172] While they are vulnerable to copying. *Directive on the Patentability 2002/0047 (COD)*, p. 2.
[173] Incremental innovation is typical of the software industry. *Directive on the Patentability 2002/0047 (COD)*, p. 6.
[174] Dreyfuss, *Expanding*, p. 26.
[175] Sub-patentable innovations are the subject-matter of the compensatory liability regime proposed by Reichman. See this study, 6.2 [4] Subject-matter of the Compensatory Liability Regime.
[176] Dreyfuss, *Expanding*, p. 26.

secret.[177] In these circumstances, second comers can easily reproduce any incremental innovation, borne on or near the face of a product, without paying appreciable costs in reverse engineering the innovator's technical know-how by proper means and without conferring any appreciable lead-time advantages on the first comer. When second comers make different selections from, and arrangements of, the software innovation contained in software products, they may avoid infringing the innovator's copyrights. Second comers can then use their versions in competition with the innovator from whom they appropriated the innovation. The vulnerability of sub-patentable (or small grain-sized) innovations to free-riding copiers under these conditions generates fears of market failure.[178]

Both the characteristics of modern software development (i.e. sequential, cumulative and incremental) and the vulnerability of sub-patentable innovations provide us the basis to evaluate the alternative proposals discussed in Chapter 6 in order to find the most appropriate sort of protection for software innovation.

---

[177] Ibid., p. 51. The Internet is often described as the biggest copy machine.

[178] Information goods have the properties of public goods. They are ubiquitous, inexhaustible, and indivisible. A second comer's use of a new information good does not diminish or exhaust it. Once disclosed to the world, anyone can use an information good without the creator's permission.

CHAPTER 3

Patent Protection of Software-related Inventions

3.1 In General

Software can be protected either by patent or copyright or both. Patent protection for software has advantages and disadvantages in comparison with copyright protection. There have been many debates concerning patent protection for software as information technology has developed and more software has been developed. These debates seem to have been caused by the characteristics of software, which is intangible and also has a great value. It takes a huge amount of resources to develop new and useful programs, but they are easily copied and easily transmitted through the Internet all over the world. Before investigating the protection of BMIs, this chapter discusses the patent protection for computer software.

Computer programs remain intangible even after they have actually come into use. This intangibility causes difficulties in understanding how a computer program can be a patentable subject-matter. There has been a presumption that a computer program is analogous to an algorithm, which has been regarded as unpatentable subject-matter. Some argue that the Agreement on Trade-Related Aspects of Intellectual Property Rights (TRIPs Agreement) requires us to grant patents for software because according to Article 27(1) of the TRIPs, patents are available for any inventions, whether products or processes, in all fields of technology, provided that they are new, involve an inventive step and are susceptible of industrial application. However, it depends on how these words are interpreted. It is controversial whether software belongs to technology or whether pure software is capable of industrial application. Although the TRIPs

Agreement is a basis for wider protection for software, the decision should be made on sound economic principles. The questions of whether and to what extent computer programs are patentable remain unresolved.[1]

TRIPs Agreement

The TRIPs Agreement was signed in the Uruguay round of the GATT negotiations and came into effect on 1 January 1995. It is the most comprehensive multilateral agreement on intellectual property (IP). It establishes for the first time a global minimum standard of intellectual property right (IPR) protection. It covers seven areas of IPRs:[2] (1) copyright and related rights; (2) trademarks; (3) geographical indications; (4) industrial designs; (5) patents; (6) lay-out designs of integrated circuits; and (7) undisclosed information or trade secrets. It specifies detailed requirements for the substantive content of national IPR legislation regarding extent of coverage, terms of protection, mechanisms of enforcement and so on. With the Agreement, the protection of IP has become an integral part of the multilateral trading system as reflected in the World Trade Organization (WTO). The purported aim of the TRIPs was to avert trade tensions by introducing more order and predictability in the system and to settle disputes more smoothly. Countries failing to comply with the TRIPs standards could be subject to trade sanctions, if the dispute settlement mechanism of the WTO has determined the

---

[1] James P. Chandler, *Patent protection of computer programs*, http://mipr.umn.edu/archive/articles/Chandler2000_01_01.htm, accessed 7 January 2001.

[2] The TRIPs Agreement integrated a number of international IPR convention, including the Paris Convention, Berne Convention and the Washington Treaty of 1989. See Rajah Rasiah, *TRIPs and Capability Building in Developing Economies*, March 2002, United Nations University, Institute for New Technologies, Discussion Paper Series 2002-1; *TRIPS – Trade-Related Intellectual Property Rights: A New Regime* www.southcentre.org/publications/trips/tripsmaintexttrans-01.htm accessed 5 June 2002; Peter Einarsson & Marie Bystrom, *TRIPS: Consequences for developing countries Implications for Swedish developing cooperation*, Consultancy Report to the Swedish International Development Cooperation Agency, August 2001, available at www.grain.org/docs/sida-trips-2001-en.pdf accessed 10 June 2002.

existence of a case of non-compliance with the TRIPs Agreement.

There is no provision in TRIPs equivalent to Article 52.2 of the EPC which provides specific exclusions from the scope of inventions, e.g. schemes, rules and methods for performing mental acts, playing games or doing business, and programs for computers.[3] However, any application for a patent covering such subject-matter must satisfy the basic criteria of novelty, inventive step and industrial applicability. In the case of computer programs, the industry has advanced to the point where software innovations are largely a mix of modifications, rearrangement or re-use of existing programs.[4] Even though a new assemblage might pass the test of novelty, it would fail the criteria of inventive step if the assemblage would be obvious to a skilled programmer.

[1] Main Issues in Computer-related Inventions

It is necessary to review the main issues in computer-related inventions to address the direction of this study. There have been many debates on whether computer software is a statutory subject-matter. Furthermore, as information technology and e-commerce develop, the patentability of methods of doing business in the Internet is becoming an important issue.

[a] Definition of Business Method Patents

Because the definitions of business methods and business method patents have been

---

[3] See this study, 3.1 [1] [b] Patentability.
[4] See this study, 2.5 Characteristics of Modern Software Development.

uncertain and divergent, the patentability of business method remains unclear. A clear and uniform definition of business method patents is required to address the scope of business method patents and to examine the patentability of business methods. Professor Ryuta Hirashima defined business method inventions as "inventions that are embodied or applied not in the industrial field in the general sense, but in commercial areas of industry centring on financial and service transactions."[5] The Japanese Patent Office (JPO) defined business method inventions as "inventions which are concerned with methods or systems of doing business using computers or the Internet".[6] Business methods can also be understood as follows:

(1) Methods or systems of doing business

(2) Computer-implemented methods or systems of doing business

(3) Internet-based methods or systems of doing business

(4) Methods or systems of doing business which is using computers or the Internet

In this study, "business methods" are defined as "methods or systems of doing business using computers and/or the Internet".

Under this definition, "business method inventions" (BMIs) can be defined as "inventions which are concerned with methods or systems of doing business which are using computers and/or the Internet". And "business method patents" (BMPs) can be defined as "patents which are concerned with methods or systems of doing business using computer and/or the Internet". BMPs mainly occur in businesses where computer applications are used to facilitate the businesses or to protect the system utilized. They

---

[5] Ryuta Hirashima, *Changes in Subject-matter under the US Patent Law*, Institute of Intellectual Property (Japan), March 2000, p. 27.

[6] Chapter 1 of *Implementing Guidelines for Inventions in Specific Fields*, JPO, 1997. http://www.jpo.go.jp/infoe/sisine.htm accessed 15 December 2001.

are patents recognizing the actualisation of information technology. In Japan they are usually referred to as "business model patents".

[b] Patentability[7]

There are five major statutory standards of patentability. To be patentable an invention must satisfy all of these standards. These are subject-matter, novelty, non-obviousness usefulness and disclosure requirements.

Subject-matter

To be patentable, an invention must be classified as having statutory subject-matter. This requirement appears in the EPC Article 52 and the Title 35 Patents of United States Code (35 U.S.C.) 101. The Article 52 lists four categories that shall not be regarded as inventions:

> *(a) discoveries, scientific theories and mathematical methods;*
> *(b) aesthetic creations;*
> *(c) schemes, rules and methods for performing mental acts, playing games or doing business, and program for computers;*
> *(d) presentations of information.*

The 35 U.S.C. 101 lists four classes of statutory subject-matter. The four classes are processes, machines, articles of manufacture and compositions. Most computer software patents fall into a process or a machine classification.

Novelty

EPC Article 54 and 35 U.S.C. 102 include the novelty requirement for patentability. The

---

[7] Chris Holt, *Patentability of Internet Business Models*, www.ukans.edu/~cybermom/CLJ/holt.html accessed 7 January 2001.

EPC Article 54 stipulates:

> *(1) An invention shall be considered to be new if it does not form part of the state of the art.*
> *(2) The state of the art shall be comprised everything made available to the public by means of a written or oral description... before the date of filing of the European patent applications.*

A prior disclosure of an invention in a publication, a prior use of an invention or prior general public knowledge of an invention prevents an invention from being granted patent protection. All the acts above join together to form the "prior art". An invention must include at least one feature that does not exist within any one reference from the body of prior art.

This novelty requirement applies to Internet business methods in the same manner. A piece of prior art, only if it includes "each and every" element of an invention, prevents the invention from being patented. A non-Internet business practice cannot be a prior art that rejects the novelty of Internet business methods.

Non-obviousness

In order for an invention to be patented, a non-obviousness requirement also must be satisfied. The EPC Article 56 provides the inventive step requirement:

> *"An invention shall be considered as involving an inventive step if ... it is not obvious to a person skilled in the art."*

35 U.S.C. 103 establishes the non-obviousness requirement of patentability. An invention must improve the prior art to the extent that the improvement is not obvious to a person skilled in the art of technology of the invention. In other words, the invention must present unexpected effects. Thus, subject-matter passing requirement under 35 U.S.C. 102 may nonetheless be unpatentable if it would have been obvious to one of

ordinary skill in the art at the time when the invention was made.

Unlike novelty analysis, a determination of obviousness may be based on a combination of references ('mosaicking'). There must be some suggestion or motivation to support the combination. Single or multiple references might be used to reject an invention on the ground of obviousness.

The statutory non-obviousness test seeks to reward inventions that have a low probability of success.[8] It influences the decisions of R&D managers to pursue or ignore specific research projects. The non-obviousness standard encourages researchers to pursue projects whose success appears highly uncertain at the outset. According to the standard, only the results from uncertain research should be rewarded with a patent. The non-obviousness requirement also guarantees that a minimum amount of information is disclosed in exchange for a patent. Non-obviousness tries to assure that patents will only be given for those inventions that would not have been made without the promise of a patent.[9] That is, society will reward only those who require a reward to do their work. Thus, the job of non-obviousness is to encourage inventions that would not otherwise be made since inventors could not recoup all the benefits of their inventions.

In Japan, non-obviousness is tested by judging whether or not the invention would have been easily created by a person skilled in the art at the time of filing. Thus, it is important to understand the ordinary creative ability of a person skilled in the art. The following are considered to be within the exercise of ordinary creative ability expected of a person skilled in the art:[10] (1) application to other fields; (2) supplement or

---

[8] Robert P. Merges, *Uncertainty and the Standard of Patentability*, http://www.law.berkeley.edu/journals/btlj/articles/07_1/Merges/.../text.htm accessed 30 October 2001.
[9] Edmund W. Kitch, *The Nature and Function of the Patent System*, 20 J.L. & Econ.265 (1977).
[10] JPO, *Examination Guidelines For Computer Software-related Inventions*, June 1996.

replacement by a commonly known means for systematization; (3) implementation by software of functions which are otherwise performed by hardware; and (4) systematization of human transactions.[11]

Disclosure and claim definiteness

In return for the patent grant, the patentee must fully disclose the invention. The EPC

Article 83 provides disclosure requirement:

> *The European patent application must disclose the invention in a manner sufficiently clear and complete for it to be carried out by a person skilled in the art.*

The EPC Article 84 provides claim definiteness requirement:

> *The claims shall define the matter for which protection is sought. They shall be clear and concise and be supported by the description.*

35 U.S.C. 112 establishes the disclosure requirements for a valid patent:

> *'The specification shall contain a written description of the invention, and of the manner and process of making and using it, in such full, clear, concise, and exact terms as to enable any person skilled in the art to which it pertains, or with which it is most nearly connected, to make and use the same, and shall set forth the best mode contemplated by the inventor of carrying out his invention.'*
>
> *'The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.'*[12]

The statute recognizes four disclosure requirements: (1) the written description requirement, (2) the enablement requirement, (3) the best mode requirement, and (4) the claim definiteness requirement.[13]

The purpose of the written description requirement is to convey with reasonable clarity to those skilled in the art that he was in possession of the invention. The

---

[11] These will be explained in detail at 3.4 Software Patents in Japan.
[12] 35 U.S.C. 112 (1999).
[13] Cohen & Lemley, *Patent Scope*.

description must clearly allow persons of ordinary skill in the art to recognize that the applicant invented what is claimed. A properly written description serves to document the inventor's conception of the claimed subject-matter on the filing date.

In contrast to the written description requirement, the main purpose of the enablement requirement is to teach the public how the invention works.[14] This is the essential trade between the society and the inventor. The inventor enables the society to practice the invention and in return the society provides the inventor with the exclusive patent rights in the invention. The patent is required to enable not the general public but only those skilled in the relevant art to make and use the invention without undue experimentation.

There are three typical issues in the context of the enablement requirement: (1) the extent of knowledge possessed by a hypothetical person "skilled in the relevant art"; (2) the time at which the specification must be enabling; and (3) the definition of "undue experimentation."[15] The specification need not disclose what is known to a person skilled in the relevant art. The specification must be enabling at the time the application is filed. A specification which becomes enabling in the light of events occurring only after filing does not satisfy the disclosure requirement. A specification which requires some experimentation may nonetheless be enabling as far as it does not require undue experimentation.

The best mode requirement is part of the deal between the society and the inventor. The best mode obligation requires that the patent applicant disclose the best way of carrying out the invention known to the inventor.

Finally, the patent applicant must abide by the claim definiteness requirement that

---

[14] Ibid.
[15] Ibid.

the claims must particularly point out and distinctly claim the subject-matter of the invention.[16] Because the patent scope is defined by the claims, the claims must clearly indicate the extent of the patentee's right of exclusion. While the written description requirement governs patentee behaviour by limiting overreaching beyond the scope of the invention, the claim definiteness requirement regulates the future behaviour of others by insisting that they receive notice of the scope of the claimed invention: Whether a claim satisfies the claim definiteness requirement turns on whether those skilled in the art would understand what is claimed, or the scope of the claim, when the claim is understood in the light of the specification.

Industrial applicability or usefulness

The EPC Article 57 provides industrial application requirement. An invention can be regarded as susceptible of industrial application if it can be used in any kind of industry, including agriculture. In short, an invention must be useful to be patentable. As long as they are functional and not aesthetic, almost all inventions are useful unless they are for illegal purposes and non-operable inventions.

Technical effect

In the early stage, the technical effect doctrine was the most widely-followed doctrine adopted by the EPO as a criterion for identifying the patentability of software-related inventions. Recently, however, the Technical Boards of Appeal of the EPO (Technical Boards) have changed their attitude to the patentability of computer programs. In *T 833/91* (Decision of 16 April 1993, Application number: EP86109711, Title of the

---

[16] Ibid.

application: 'Simulation of computer program external interfaces'),[17] the Technical Boards concluded that the technical contribution to the art might exist *either* in the underlying problem and solution of the claimed invention, *or* in the means constituting the solution of the underlying problem, *or* in the effects achieved by the solution of the underlying problem.[18]

The *Directive*[19] on the patentability of computer-implemented inventions, defines "technical contribution" as to mean a contribution to the state of the art in a technical field which is not obvious to a person skilled in the art (Article 2). The *Directive* makes it clear that the "technical contribution" should be assessed not in the context of novelty but under inventive step (Article 3), and that the "technical contribution" should be assessed by consideration of the difference between the scope of the patent claim as a whole (comprising both technical and non-technical features), and the state of the art (Article 4).

[c] Prior Art

Without properly established prior art it is impossible to search for prior art and to examine patent applications properly. Due to the characteristics of software, the patent protection for software has been limited. This limitation resulted in insufficient published prior art in this field. Inadequate prior art causes bad patents. Invalid patents

---

[17] Not published in the Official Journal EPO. http://www.softwarepatenter.dk/ekstern/t/t910833eu1.htm accessed 15 July 2002; http://legal.european-patent-office.org/dg3/pdf/t920077eu1.pdf accessed 15 July 2002.

[18] Robert Hart, Peter Holmes, John Reid, Study Contract ETD/99/B5-3000/E/106: *The Economic Impact of Patentability of Computer Programs,* the study was conducted by the Intellectual Property Institute, London and finalized in March 2000 (afterwards, Robert Hart, *The Economic Impact of Patentability of Computer Programs*). www.europa.eu.int/comm/internalmarket/en/intprop/indprop/softpaten.htm accessed on 26 October 2000.

[19] *Directive on the Patentability 2002/0047 (COD)* (20 February 2002).

may impose social costs. There have been persistent arguments that patents in the software area, especially those regarding business methods, are of poor quality. Because methods of doing business exist mainly in practice and they have not been patented in the past, there is very little readily accessible prior art of business methods. To solve these prior art and patent quality problems, suggestions have been made about how to improve prior art information databases.[20]

In addition to the poor database, the issues of when the prior art was disclosed and what makes prior art available on the Internet, have not yet been determined.

[2] Theories Concerning Software-related Inventions

Because the subject-matter of BMPs usually fall within the scope of computer software, in order to understand the patentability of business methods fully it is necessary to review cases and theories concerning software-related inventions. Because of the uncertainty of patentability, a number of theories have been applied to software-related inventions as the basis for denial.

[a] The Mathematical Algorithm Exception

A mathematical algorithm is a procedure for solving a mathematical problem.[21] The term "mathematical algorithm" was defined as a process that merely expresses a mathematical principle in the language of a computer program.[22] If the core component

---

[20] Amazon.com's Jeff Bezos suggested that the USPTO provide pre-grant oppositions before the issuance of a business method or software patent. Merges, Dreyfuss, Reichman, and Nard have suggested that opposition be provided.

[21] *Diamond v. Diehr*, 450 U.S. 175, 186 (1981); *Gottschalk v. Benson*, 409 U.S. 63, 65 (1972).

[22] The *Benson*, 409 U.S. at 65.

of a computer program is no more than an expression of a mathematical algorithm, it is not patentable subject-matter because such an algorithm is an expression of a fundamental scientific principle, which is similar to a law of nature.[23]

The mathematical algorithm exception is an attempt to prevent a patent applicant from pre-empting an abstract principle of human knowledge. To allow such an abstract principle to be pre-empted would completely imbalance the trade-off between (inventor's) incentive and (the public's) access. This gravely impedes future innovation. However, a patent on the application of a mathematical algorithm to a computer (or to some specialized machine) does not preclude other from applying the same mathematical algorithm to achieve other result in a computer (or specialized machine). It is the application of the mathematics for a narrow practical use, not the mathematics itself.[24]

In 1939, the US Supreme Court ruled that a mathematical algorithm was not patentable subject-matter.[25] The court stated that "while a scientific truth, or the mathematical expression of it, is not a patentable invention, a novel and useful structure created with the aid of knowledge of scientific truth may be".[26] In *Gottschalk v. Benson,*[27] the court responded in the same way as the court had in 1939 in the *Mackay Radio and Tel. Co.* The *Benson* court announced that a patent should not be granted for any software invention which pre-empted a mathematical algorithm. United States Patent and Trademark Office (USPTO) has used this mathematical algorithm doctrine to reject many software patent applications. However, Court of Appeals of the Federal

---

[23] The *Diehr*, 450 U.S. at 186; the *Parker*, 437 U.S. at 593.
[24] Kenneth W. Dam, *Intellectual Property in an Age of Software and Biotechnology,* Chicago Working Paper in Law & Economics, May 1995.
[25] *Mackay Radio and Tel. Co. v. Radio Corp. of Am.*, 306 U.S. 86, 94 (1939).
[26] Id.; *Diehr*, 450 U.S. at 188; *Parker*, 437 U.S. 584, 591; *Benson*, 409 U.S. at 67; *In re Abele*, 684 F.2d 902, 907 (CCPA 1982); *In re Walter*, 618 F.2d 758, 763 (CCPA 1980).
[27] 175 USPQ 673 (1972 U.S. Supreme Court).

Circuit (CAFC) in the *State Street Bank & Trust Co. v. Signature Financial Group, Inc.*, (*SSB*)[28] held that statutory subject-matter of mathematical algorithms is decided according to whether such calculations produce 'a useful, concrete and tangible result'.

Recently, the Commission of the European Communities (European Commission) issued a Directive,[29] in which the patentability of algorithms has been explained. According to the explanation, the term "algorithm" may be understood to mean any sequence of actions intended to accomplish a specific task. The mere existence of an algorithm does not make an invention non-patentable. An algorithm may exist in a computer-implemented invention, an invention relating to a conventional machine or the process carried out by that machine. An algorithm may be used in many different functions and may achieve different effects. An algorithm which is regarded as a theoretical entity without connection with a physical environment is inherently non-technical. A patent claim to an invention based on a particular algorithm would not extend to other applications of that algorithm.

[b] The Mental Steps Doctrine

Processes involving mental operations, such as "selecting", "determining" and "observing", were considered unpatentable. The rationale for the doctrine was to ensure that a patentee could not obtain a monopoly on thought processes. This was based on the idea that, since individual freedom of thought and protection of the creations of the mind of the individual for the benefit of the individual are paramount values, they should always be preserved for the good of all in human society. According to this

---

[28] 149 F.3d 1368 (Fed. Cir. 1998).
[29] *Directive on the Patentability 2002/0047 (COD)*, p. 7.

doctrine a scientific concept or mere idea cannot be the subject of a patent. The courts concluded that processes involving mental steps could not be patented because restrictions on mental processes would not serve to promote the progress of technology. To be patentable, such a process should be applied to transform objects in the real world.[30]

In re Abrams[31] endorsed a set of rules by which to judge processes involving mental steps. The first rule is that if a process is "purely mental" it is not patentable. The second is that if a process has both mental and physical steps, but the advance over the prior art is found only in the mental steps, it is not patentable. The third is that if a process claim has both mental and physical steps and there is novelty in the physical as well as in the mental steps, then the process is patentable. In *In re Prater,*[32] the examiner rejected the claims on the ground that, if the novel part of the process was a mental step and non-patentable subject-matter, the whole process was unpatentable. The court reasoned that the inventions in the *Abrams* and the *Prater* are different due to the fact that the *Abrams* process could only be performed in the mind and the *Prater* process did not require any steps to be performed in the human mind. The *Prater* court held that the process was patentable because it was a useful art and capable of being performed without human intervention. The court held that its patentability was not precluded because it could be performed by mental steps alone.[33] The USPTO petitioned for a rehearing because it believed the decision opened the door for the patentability of computer programs. The *Prater* court reversed itself. The court stated that the claim could be interpreted as a process that could be performed in the human

---

[30] James P. Chandler, *Patent protection of computer programs.*
[31] *Abrams*, 188 F.2d at 166.
[32] 415 F.2d 1378 (CCPA 1968).
[33] 415 F.2d at 1389.

brain or by hand with pencil and paper. The *Prater* decision in the rehearing had importance in that it elevated the protection of society's freedom of thought to the same level as the protection of the individual's invention for his exclusive benefit.

The mental steps doctrine, however, was denied by the CCPA in *In re Musgrave.*[34] The court stated that the fact that the process could be carried out in the human mind, or required the person performing the process to think, did not preclude patentability.[35] A process claiming only mental processes or calculations is not patentable subject-matter because it claims to pre-empt the use of the human brain.[36]

But the Supreme Court's reference to "mental processes" in *Gottschalk v. Benson* (1972)[37] made it unclear whether the doctrine has been entirely eliminated.

[c] The Printed Matter Exception

The printed matter exception is a judicially created doctrine that holds that an invention consisting of "the mere arrangement of printed matter on a sheet or sheets of paper, book form or otherwise, does not constitute patentable subject-matter.[38] Printed matter can be considered to be an abstract idea and therefore unpatentable subject-matter, since it does not interact with the real world of machines, processes and manufactures. The term "printed matter" comprehends any intangible expression that has neither functional manifestation nor a novel relation to structure.

Some early cases supported the patentability of printed materials.[39] In *Cincinnati*

---

[34] 431 F.2d 882 (CCPA 1970).
[35] 431 F.2d 892.
[36] See 431 F.2d 889-90.
[37] 175 USPQ 673 (1972 U.S. Supreme Court).
[38] Provided in section 4886 of the Revised Statutes, 35 USCA 31. *In re Russell*, 48 F.2d 668 (CCPA 1931). Also see *In re Dixon*, 44 F.2d 881 (CCPA 1930).
[39] See *Cincinnati Traction Co. v. Pope*, 210 F. 443 (6th Cir. 1913).

Traction Co. v. Pope, the court ruled that "transfer tickets" for use by street railway companies involved patentable subject-matter. The court stated that the claimed invention involved the physical structure of the ticket, and that the presence of printed matter such as "conventional indications" could not deny its patentability.

This doctrine came into its current form in *In re Russell.*[40] The *Russell* invention is relevant to a method of making names easier to look up in phone directories. The method consists of arranging the names in both alphabetical and phonetical order. The court stated that the mere arrangement of printed matter on a sheet of paper does not constitute a manufacture.[41] Since it is not a manufacture, the invention does not fall within the scope of patentable subject-matter. In the *Dixon*[42] a lawyer's agreement form which contained a lawyer fee clause was found to be unpatentable. Further cases include *Kieferle v. Kingsland;*[43] *Wier v. Coe;*[44] *In re Lockert;*[45] *In re Reeves;*[46] and *In re Sterling.*[47]

[d] The Business Method Exception

Because of their intangible characteristics, business methods by themselves were considered as outside the scope of patentability. The exception excluded business plans or systems from patent protection. A claim to the steps needed to complete a business process might be rejected by the exception. This is a judicially created exception to statutory subject-matter.

---

[40] 48 F.2d.
[41] The *Russell* at 669.
[42] 44 F.2d 881 (CCPA 1930).
[43] 79 F.Supp. 700 (D.C. 1948).
[44] 33 F.Supp. 142 (D.C. 1940).
[45] 65 F.2d 159.
[46] 62 F.2d 199.
[47] 70 F.2d 910.

3.2 Software Patent in the US

[1] Introduction

Under the US patent statute, a patent may be granted for any new and useful process, machine, manufacture or composition of matter. This broad statutory definition of subject-matter has enabled the US to extend the scope of patentable subject-matter. The US has thus developed criteria for patentability of computer-related inventions.

[2] Statutory Patentability Development

The Patent Act of 1793[48] defined statutory subject-matter as *'any new and useful art, machine, manufacture or composition of matter, or any new or useful improvement thereof'*. The Patent Act of 1952 replaced 'art' in the Patent Act of 1793 by 'process'. The Committee Reports of the 1952 Act noted that *"statutory subject-matter includes anything under the sun that is made by man".*[49] To promote innovation by balanced patent protection, the Intellectual Property and Communications Omnibus Reform Act of 1999 (29 November 1999) was suggested.

In the US, an abstract idea, mental steps, scientific truths and mathematical expressions are considered as non-patentable. An idea itself is not patentable, because an abstract idea, a theory and a plan do not produce any physical effect even after they are actually operated. An invention employing a mental step, as an essential component of a process cannot be patented. Novel and useful apparatus or processes created by

---

[48] This was authored by Thomas Jefferson.
[49] *Report of the President's Commission on the Patent System* (US 1966).

application of the scientific truth are patentable. The mere use of scientific truth, or the mathematical expression of it, however, excludes an invention from being qualified as a patentable invention.

In response to the decisions of the CAFC, particularly *In re Alappat*[50] and *In re Lowry*,[51] the USPTO issued the Examination Guidelines for Computer-related Inventions in 1996 (1996 Guidelines). By the 1996 Guidelines, computer-related inventions came to include inventions implemented in a computer and inventions employing computer readable media. These Guidelines recognized as patentable recording media carrying programs and thus facilitated the granting of more software-related patents. "The real question is now no longer whether software-related inventions claim patentable subject-matter, but rather whether such inventions are novel and non-obvious."[52] These 1996 Guidelines stipulate statutory and non-statutory subject-matter as follows:

Statutory subject-matter

- a machine ; a computer or other programmable apparatus whose actions are directed by a computer program or other form of 'software'.
- an article of manufacture ; a computer-readable memory that can be used to direct a computer to function in a particular manner when used by the computer.
- a process ; a series of specific operational steps to be performed on or with the aid of a computer.

Non-statutory subject-matter

- a compilation of arrangement of data, independent of any physical element.

---

[50] 33 F.3d 1526 (Fed. Cir. 1994).
[51] 32 F.3d 1579 (Fed. Cir. 1994).
[52] J Fellas, *The Patentability of Software-related Inventions in the United States.* European Intellectual Property Review, 330, 1999, at 333.

- a known machine-readable storage medium that is encoded with data representing creative or artistic expression (e.g. a work of music, art or literature).

- a data structure independent of any physical element.

- a process that does nothing more than manipulate abstract ideas or concepts.

[3] Case Law Developments

The US case law and the USPTO practice regarding software patenting have been vague, constantly in flux and inconsistent.[53] The early case law on software patents and 35 U.S.C. 101 created significant barriers to patentability.

In recent years, the software patent case law and the USPTO practice have indicated an increased willingness to allow software to be patentable. With these changes, the current issues on the software patents become "to what extent has the patentability of software been extended?" and "what kind of software inventions can be regarded as patentable subject-matter?"

[a] The Early Software Patent Case Law

Gottschalk v. Benson (1972): *the first modern software patent decision*

In 1972, the Supreme Court decided its first case dealing directly with the issue of software patentability.[54] The *Benson* invention related to the conversion of "binary coded decimal" numerical information into binary numbers. The USPTO had rejected

---

[53] Keith E. Witek, *Developing a Comprehensive Software Claim Drafting Strategy for U.S. Software Patents*, http://www.law.berkeley.edu/journals/btlj/articles/11_2/Witek/html/text.html accessed 21 April 2001.

[54] Witek, *Comprenhensive Software Claim.*

the claims stipulating that the subject-matter was non-statutory because it involved a "mental process" and a "mathematical step".

The CCPA, however, argued that "*very little remains of the mental steps doctrine*". The CCPA concluded that the process could be carried out without any involvement of a human being once the apparatus was set up. The court argued that all the operational steps in the claim were designated and human judgment or decision was not required.[55] The court reversed the rejection by the Board unanimously and the USPTO filed a writ of certiorari to review this judgment and the Supreme Court granted it.[56]

Reversing the decision of the CCPA, the Supreme Court stated that the mathematical procedures could be carried out either in an existing computer, or without a computer. The Court held that the mathematical formula for converting binary decimal code to pure-binary in this case has no substantial practical application except in connection with a digital computer, which means that the patent will be a patent on the algorithm itself and will wholly pre-empt the mathematical formula.[57]

On the other hand, refusing the request by the hardware manufacturers who wanted the Court to declare that all software was unpatentable, the Court remarked that the decision did not intend to preclude patent protection for programs operating a computer. However, in the *Flook*, the progress of software patenting seemed to worsen.

Parker v. Flook[58]

The invention in this case was a computerized method of continuously updating alarm limits during a chemical refining process.

---

[55] H.W.A.M. Hanneman, *The Patentability of Computer Software*, Kluwer Law and Taxation Publishers, 1985 (afterwards, H.W.A.M. Hanneman), p. 49.
[56] Writ of certiorari (*Benson and Tabbot*) granted, 172 USPQ 577.
[57] 175 USPQ 673 p. 676.
[58] 473 U.S. 584 (1978).

The Supreme Court held that it was unpatentable because the only novel feature of the process was a computer program. The court reasoned that if it ignored the mathematical algorithm, updating the alarm limit, the claimed invention contained nothing new or inventive.[59] The court noted that the claims should be considered without the mathematical algorithms to determine whether patentable subject-matter remained. Therefore the issue was to decide whether post-solution applications of the formula made the *Flook* method patentable. Although the claim did not wholly pre-empt the mathematical formula, insignificant post-solution activity cannot transform an unpatentable principle into a patentable process. The Supreme Court concluded that the *Flook* process was unpatentable, not because it contained a mathematical algorithm, but because once that algorithm was assumed to be within the prior art, the application did not contain any patentable invention.

The minority, however, criticized the majority for invoking the criteria of novelty and inventiveness under 35 U.S.C. 101.[60]

This case appears to be incompatible with the following *Diehr* discussed below.[61] The court itself mentioned that the *Flook* claim did not pre-empt the formula itself, because the uses of the formula outside the petrochemical and oil-refining industry remained in the public domain. (A patent with the *Flook* claim would not monopolize the uses of the formula without any subsequent adjustment of a valve as in the *Flook* invention.)[62] Because the *Flook* claim did not wholly pre-empt the mathematical algorithm, if the post-solution, the updated alarm-limit value, had been recognized as significant, it could have been patentable subject-matter.

---

[59] 473 U.S. at 589-91.
[60] H.W.A.M. Hanneman, pp. 66-67.
[61] Minoru Sano, *Software and Intellectual Property*, published in Japanese by Iwanami Shoten, Publishers, Tokyo 1997.
[62] H.W.A.M. Hanneman, pp. 66-70.

The *Flook* decision left uncertain the inquiry of subject-matter patentability of software inventions involving mathematical algorithm, because the court examined the claims without the mathematical algorithm in determining whether the patentable subject-matter remains.

Diamond v. Diehr (1981)[63]

Even though it culminated in the *Flook*, the rising tide against the patentability of software started to recede with the *Diehr*. In this case the US Supreme Court granted a patent on a computer-related invention for the first time. The *Diehr* court stated that claims must be considered as a whole to determine patentability, and that claims may not be considered unpatentable simply because they contained algorithm. The *Diehr* court rejected the *Flook* analysis which dissected the claims into old and new elements, ignored the presence of the old elements and attacked each element in isolation. The *Diehr* court explained that claims have a synergistic effect, and that the novelty of any process does not influence subject-matter patentability. The court held that "a claim drawn to subject-matter otherwise statutory does not become non-statutory simply because it uses a mathematical formula, computer program, or digital computer".[64] In the light of the *Diehr* (as well as the *Flook*), an invention employing mathematical algorithms is patentable subject-matter if it does not pre-empt the algorithms.

However, the *Diehr* court reaffirmed that mathematical algorithms are not patentable as such. It suggested the necessity of other limitation that renders the software claim patentable as a whole. The court stated that merely limiting an algorithm or software innovation to a specific "technological environment" or adding insignificant

---

[63] 450 U.S. 175.
[64] 450 U.S. 187.

post-solution activity to software claim would not be enough to be patentable subject-matter.[65] Thus, pulling back from the *Flook*, the *Diehr* court made it clear that significant additional physical limitations over software algorithms must be included in the claim to render the software invention patentable.[66]

Because the *Diehr* failed expressly to overrule the *Benson* or the *Flook* decisions, the *Freeman-Walter-Abele* test (FWA test)[67] remained based on the mathematical algorithm approach. However, the *Diehr* decision significantly influenced on the attitude of the USPTO for the patentability of computer-related inventions. After the decision the USPTO took a more favourable approach towards these inventions, e.g. issuing guidelines for the examination of computer-related inventions. These approaches resulted in significant drop in the number of appeals to the CCPA (or the CAFC).[68]

The *Diehr* invention[69] was essentially quite similar to the *Flook* invention. Both processes involved an initial calculation, continual remeasurement and recalculation, and certain controlling by using the value obtained from the calculation. The drafting methods of the claims to the two inventions were regarded as different each other. It appears the difference in drafting claims made the one patentable and the other unpatentable. If the claims in the *Diehr* invention had been drafted in the style employed by the *Flook* application, would the claims have been held as non-statutory? And if the claims in the *Flook* invention had been drafted in the style employed by the

---

[65] 450 U.S. 191-192.
[66] 450 U.S. 192.
[67] This FWA test was named by the cases concerned. It was set forth in the *Freeman*, modified by the *Walter* and interpreted in the *Abele*. The test will be explained in detail later.
[68] H.W.A.M. Hanneman, p. 91.
[69] The actual invention is continuous monitoring of temperature and real time recalculation of the cure time. The use of the Arrhenius equation was not novel in the art of rubber curing. See Dennis S. Karjala, *The Relative Roles of Patent and Copyright in the Protection of Computer Programs*, 17 John Marshall Journal of Computer and Information Law 41 (Autumn 1998).

Diehr, would the claims have been held as statutory? If the answer is "yes", it may have the consequence that the patentability of an invention depends on patent claim drafting as well as the invention itself. These questions in the two cases were explained in the *Abele* case (1982) by the FWA analysis. To examine more effectively the patentability of software involving a mathematical algorithm, the CCPA introduced the FWA test.

The *Freeman-Walter-Abele* test

The USPTO rejected the *Freeman* claim on the ground that the claim constituted mathematical algorithm. Criticizing the decision of the USPTO, the court created a two-part test for analysing whether a claim pre-empts an algorithm.[70]

(1) First, determine whether the claim directly or indirectly recites an "algorithm".

A claim which does not recite an algorithm cannot wholly pre-empt an algorithm.

(2) Second, analyse the claim to determine whether in its entirety it wholly pre-empts that algorithm.[71]

The court emphasized that the *Freeman* claim did not recite a mathematical algorithm because the claims did not recite any mathematical calculations, formulae, or equations.

In the *Walter* the CCPA restated the second step of the *Freeman* test. The *Walter* analysis has two steps: (1) determine whether the claim directly or indirectly recites an algorithm. (2) Analyse whether the algorithm is implemented in a specific manner that defines a structural relationship between the physical elements of the claim or limits the claim step. The court stated that it is appropriate to modify the terms of the second step of the *Freeman* test according to the spirit of the *Flook*.

---

[70] 197 USPQ 464 p.471.
[71] 197 USPQ 464 p.1246.

In *In re Abele* the CCPA refined this two-step test by clarifying the second step.[72] The court stated that if the claim was "otherwise statutory", although inoperative or less useful without the algorithm, the claim as well presented statutory subject-matter. The test was refined as follows: (1) determine whether the claim directly or indirectly recites a "mathematical algorithm" or "formula"; and (2) if the claim without the mathematical algorithm or formula is statutory subject-matter (i.e. an apparatus or process), then the whole claim might still present statutory subject-matter. The *Abele* court concluded that, although the algorithm did not necessarily refine or limit the earlier steps of production and detection, the discovery of an application of an algorithm to process steps of an overall process which is statutory constituted statutory subject-matter. According to this analysis, even though an invention contains a mathematical algorithm, if the claim without the mathematical algorithm is statutory subject-matter, then the claim is statutory.

The *Abele* court applied the test to the two conflicting cases, *Parker v. Flook* and *Diamond v. Diehr*. According to the *Abele*, in the *Flook*[73] if an invention did not explain how to select any of the variables used in the algorithm, no process other than the algorithm was present and no process steps to which the algorithm could be applied were present. However, in the *Diehr*[74] invention, if the claim were read without the algorithm, the process would still be a process for curing rubber, although it might not work as well since the in-mold time would not be as accurately controlled.[75]

In applying the FWA test, the CAFC had been inconsistent especially in the 1994 algorithm-related cases. The CAFC applied the test to *In re Schrader* and *In re*

---

[72] 684 F.2d 902 (CCPA 1982).
[73] 473 U.S. 584 (1978).
[74] 450 U.S. 175 (1981).
[75] *Abele*, 684 F.2d. at 907.

Warmerdam to reject the subject-matter patentability of the inventions. On the other hand, the court did not apply the test to *In re Alappat* and *In re Lowry*.

[b] The CAFC's Algorithm-related Cases

In re Schrader (13 April 1994)[76]

The claimed invention was a method useful for auctioning mixing many related items in real time.[77] The invention involved an algorithm that decided how to divide land best to maximize the return to the seller.

The CAFC held that the algorithm failed the FWA test. The court found that even though no mathematical formula was explicitly stated in the claims, the method implied the use of a simple arithmetic method without physical elements or process steps. The court ruled that the algorithm was not sufficiently connected to something in the real world, and that the program described the solving of a mathematical problem, that is, determining the optimal combination of bids.

In re Alappat (1994)[78]

The invention involved a way of generating smooth waveforms on the screen of an oscilloscope. The oscilloscope screen presents an array of pixels. Columns represent time periods, and rows the amplitude. Each input signal was sampled at prescribed times, and the detected amplitudes were stored. Each amplitude was displayed at the

---

[76] 22 F.3d at 297 (13 April 1994).

[77] *In re Schrader*, 22 F.3d. at 291. The claimed algorithm was a method for determining the combination of auction bids to be selected as winners for a variably divisible article for sale, in this case a parcel of land that could be subdivided. The algorithm analysed all bids submitted and determined which group of bids to select as 'winners' so that: (1) none of the selected bids were for overlapping subdivisions, and (2) sales proceeds for the aggregate parcel of land were maximized.

[78] 33 F.3d 1526 (Fed. Cir. 1994).

appropriate time, and straight lines connected the resulting points. The invention varied the intensity of the pixels. The intensity of each pixel is stored in a four-bit code, permitting 16 levels of intensity. The effect of the invention is a clearer picture.

While the USPTO Board applied the FWA test to the process and found it non-statutory, the CAFC reversed the rejection. The court stated that the US Supreme Court did not intend to introduce a fourth "mathematical algorithm exception". The CAFC argued that the claim as a whole is not an intangible mathematical concept, but a combination of interrelated electronic elements, which forms a machine for converting waveform data into pixel illumination data that results in the clearer image. The court added that a general-purpose computer in effect becomes a special-purpose computer once it is programmed to perform particular functions.[79]

The court held that the invention constituted a patentable practical application of a mathematical algorithm, because it produced a useful, concrete and tangible result, which is the smooth waveform. The *Alappat* court opened a new era in software patents. It established that a mathematical algorithm becomes patentable subject-matter when the algorithm is programmed into a general-purpose computer.[80] The *Alappat* case also provided a means of direct enforcement for process patents by recognizing claims involving associated hardware (a computer) or computer readable media.

In re Warmerdam (11 August 1994)[81]

The claimed invention involved technology called "bubble" systems, which are used to avoid collisions between moving objects such as mobile robotic machines. The invention was an improved bubble-bursting system that generated the bubbles by top-

---

[79] 33 F.3d 1545.
[80] See, Cohen & Lemley, *Patent Scope*.
[81] 33 F.3d 1354 (Fed. Cir. 11 August 1994).

down and bottom-up bubble generation methods. The invention also included the computer system and data structure that were used to implement the invention.

The CAFC sustained the USPTO rejection. The court concluded that the bubble generation claims were unpatentable because they failed both steps of the FWA test. Although inventions using mathematical algorithms were patentable if they produced sufficient physical results, Judge Plager stated, the claim language in the application described only an idea. Warmerdam argued that the bubble generation claims required physical measurements of the object's shape, and that measuring the contours of the object was a sufficient pre-solution activity. The CAFC dismissed this argument and stated that a data gathering step was insufficient to provide patentability.

In re Lowry (26 August 1994)[82]

The CAFC did not apply the FWA test and allowed the data structure claim in the *Lowry*, which was denied in the *Warmerdam*.[83] The invention related to an application program that managed database information using software data structures called attribute data objects (ADOs). It was claimed that using ADOs provided more efficient methods for storing, retrieving, adding, and removing information from a database.[84]

The CAFC held that all of the *Lowry* data structure claims were patentable. The CAFC argued that the data structure in the *Lowry* claims did sufficiently interact with memory to meet the functional relationship standard. The court remarked that storing data structures in memory created a physical organization of that memory and caused more efficient computer operation, which verified the existence of a functional relationship between a data structure and memory, even though physical changes could

---

[82] 32 F.3d 1579 (Fed. Cir. 1994).
[83] 33 F.3d 1354 (Fed. Cir. 1994).
[84] *In re Lowry*, 32 F.3d 1579-1581.

not be visibly detected.[85]

Both of the inventions in the *Lowry* and the *Warmerdam* claimed software data structures. The *Warmerdam* court ignored the *Alappat* decision. However, two weeks later the *Lowry* followed the *Alappat* court's spirit. These inconsistencies are found until the *SSB* case.

The *Lowry* decision is important in that it expanded the patentability of data structures from hardware to pure software inventions.[86]

In re Beauregard (1995)[87]

Considering the CAFC's five 1994 algorithm-related cases, which were *In re Shrader, In re Alappat, In re Warmerdam, In re Lowry* and *In re Trovato,*[88] to be patentable a claim must include a structural element to assure that it is focussed on more than just algorithm. Only when the claimed invention is practised through the structure included in the claim, an infringement can exist. Software inventors cannot accuse directly against producers and distributors of software because such software is infringed only by the end-users.

IBM attempted to give patentability to a claim through structural limitations that would pursue an infringement suit directed to the producers and distributors.[89] IBM claimed the invention as an article of manufacture comprising a storage device, e.g. floppy diskette or CD-ROM, encoded with machine-readable software code for

---

[85] *The Patentability of Software Data Structures after Lowry and Warmerdam,* www.nesl.edu/lawrev/vol32/vol32-3/chan.htm accessed 14 January 2001.

[86] Ibid.

[87] Shawn McDonald, *Patenting Floppy Disks, or How the Federal Circuit's Acquiescence has Filled the Void Left by Legislative Inaction,* 3 Va. J.L. & Tech. 9 (Fall 1998) 1522-1687 / © 1998 Virginia Journal of Law and Technology Association. http://vjolt.student.virginia.edu/graphics/vol3/vol3_art9.html accessed 14 December 2001.

[88] 42 F.3d 1376 (Fed. Cir. 1994).

[89] See the *IBM* cases, i.e. *T 0935/97-3.5.1* [1999] R.P.C. 861 and *T 1173/97-3.5.1*, 1999 OJ EPO [609].

implementing an algorithm. The main issue was whether a floppy disk containing the implementing software code was eligible for patent protection. The USPTO rejected the claims as non-statutory by classifying the software code contained in the storage medium as printed matter. The USPTO Board of Appeals reasoned that the novel algorithm was encoded in a prior art software language, and that the method of storage (the floppy disk) was also prior art, and therefore the sole novelty was similar to the expression in the language of software code. The Board found no novel relationship between the software code and the substrate, and no patentable invention.

The CAFC invalidated the rejection and remanded the case to the USPTO Board of Appeals. The Commissioner of the USPTO stated that computer programs embodied in a tangible medium, such as floppy diskettes, were patentable subject-matter under 35 U.S.C. 101. *Beauregard*-type article of manufacture claims can be asserted directly against the producer or distributor of infringing software. The activities of the end user no longer constitute direct infringement. The patentees are free to sue for direct infringement against the software producers or distributors without additional burdens of proof. This results in the same measure of patent protection on software as on hardware. While *Beauregard*-type claims have advantages such as easy enforcement, those claims are inconsistent with the case law on algorithm inventions in that they are, in practical effect, claims to the algorithm itself and therefore they violate the basic policy against patents on algorithms as such.

The USPTO is currently allowing article of manufacture claims having no limitations of the implementing hardware. The scope of these claims, however, is uncertain in terms of 35 U.S.C. 112. Guidance from the legislature would be a better

solution.[90]

[c] Recent Cases

Recently the CAFC made two landmark decisions, the *SSB*[91] and *AT&T Corp. v. Excel Communications, Inc* (*AT&T*),[92] which clarified the patentability of inventions involving mathematical algorithm or methods of doing business. The *SSB* drew fresh attention from businesses such as banking, securities, funding, etc. The *SSB* ruling was reaffirmed and strengthened in the *AT&T* case. The *AT&T* court noted that physical transformation was not an indispensable requirement for a claim involving a mathematical algorithm to be patentable, but merely one example of how a mathematical algorithm may bring about a useful application. Due to the two cases, the patentability of inventions involving a mathematical algorithm can be determined not by physical transformation, but by whether the algorithm has been applied in a useful way.

The SSB case (1998)

The *SSB* case cleared the uncertainties of the mathematical algorithm and the FWA test. The patent (US Patent 5,193,056)[93] covered a "hub and spoke" system of

---

[90] Richard H. Stern, *Solving the Algorithm Conundrum: After 1994 in the Federal Circuit Patent Laws Needs a Radical Algorithmectomy*, 22 AIPLA Q.J. 167, 169 (1994).

[91] 149 F.3d 1368 (Fed. Cir. 1998).

[92] 172 F.3d 1352 (Fed. Cir. 1999).

[93] Claim 1 of the patent US 5,193,056 reads:

1. A data processing system for managing a financial service configuration of a portfolio established as a partnership, each partner being one of a plurality of funds, comprising:
(a) computer processor means for processing data;
(b) storage means for storing data on a storage medium;
(c) first means for initializing the storage medium;
(d) second means for processing data regarding assets in the portfolio and each of the funds from a previous day and data regarding increases or decreases in each of the funds, assets and for allocating the percentage share that each fund holds in the portfolio;
(e) third means for processing data regarding daily incremental income, expenses, and net realized gain

administrating multiple mutual funds using computers. The "hub and spoke" system provided for the pooling in an investment portfolio to take advantages of efficiencies of scale. Because small mutual fund operators could not have such efficiencies by themselves, they participated this system paying licence fee.

The US District Court for the District Massachusetts held the patent invalid on the subject-matter ground. The court found that although the claims were written in "means for" apparatus format, they in realty covered the underlying process. The court held that the process involved a mathematical algorithm which did not satisfy the FWA test.

The CAFC reversed this decision by applying the same approach used in the *Alappat*. The CAFC stated that the critical issue was whether the algorithm had been applied in a useful way, that is to produce "a useful, concrete and tangible result". The court stated that because the transformations did produce such a result, the patent involved patentable subject-matter. The court declared that in this analysis the FWA test has little, if any, applicability to determining the presence of statutory subject-matter. The *SSB* court also resolved the *Alappat* issue concerning the patentability of a specially programmed computer, by finding that such systems are patentable subject-matter only when they produce a useful, concrete and tangible result. The *SSB* case has removed both the mathematical algorithm exception and the business method exception.

The AT&T case

The patent in the *AT&T* case related to a data processing algorithm, which was a method for "generating a message record for an interexchange call" and recording to whom the

---

or loss for the portfolio and for allocating such data among each fund;
(f) fourth means for processing data regarding daily net unrealized gain or loss for the portfolio and for allocating such data among each fund; and
(g) fifth means for processing data regarding aggregate year-end income, expenses, and capital gain or loss for the portfolio and each of the funds.

call should be billed. The district court held that the method of claims included a mathematical algorithm.[94]

The CAFC held that the invention was a process under 35 U.S.C. 101, strengthening the reasoning of the *SSB*. The court noted that since the process of manipulation of numbers is a fundamental part of computer technology, the legal boundaries placed on computer technology should be reassessed in the light of the advances of the technology. *Excel Communications* argued that the method claims constituted unpatentable subject-matter because there was no physical transformation of the data from one state into another. The CAFC, however, noted that physical transformation is not an indispensable requirement for a claim involving a mathematical algorithm to be patentable, but merely one example of how a mathematical algorithm may cause a useful application. The court stated that the mathematical algorithm of the claims was applied in a practical manner to produce a useful result. The *AT&T* case is said to represent the demise of the mathematical algorithm exception.

3.3 Software Patents in the EPO

[1] Introduction

According to Article 52(1) of the EPC, European patents shall be granted for any inventions which are susceptible of industrial application, which are new and which involve an inventive step. In Article 52(2), the EPC provides a list of subject-matter which will not be regarded as patentable. In the context of software (and business method) patents, the most important prohibitions are: (a) discoveries, scientific theories

---

[94] 172 F.3d 1355 (Fed. Cir. 1999).

and mathematical methods; (c) schemes, rules and methods for performing mental acts, playing games or doing business, and programs for computers; and (d) presentation of information.

Although it would appear from a plain reading of Article 52(2) that software patents are unattainable in Europe, Article 52(3) reads:

> *The provisions of paragraph 2 shall exclude patentability of the subject-matter or activities referred to in that provision only to the extent to which an European patent application or European patent relates to such subject-matter or activities* ***as such.***

In addition, in order to be patentable Rules 27 and 29 require that "an invention must be of a technical character to the extent that it relates to a technical field, must be concerned with a technical problem and must have technical features in terms of which the matter for which protection is sought can be defined in the patent claim".

Although methods for doing business or programs for computers are *as such* explicitly excluded from patentability, inventions which have a technical character may be patentable, even if the claimed subject-matter defines or involves a business method or a computer program. The term, "as such" is of importance. Patents can be granted for inventions which have a 'technical character'. 'Technical character' can be interpreted as requiring both that an invention must belong to a field of technology and that the invention must also make a technical contribution to the technological state of the art. Conversely, "computer programs as such" are excluded from patentability due to their having no technical character.[95] Patents can only be granted for inventions which are new, inventive and industrially applicable. The Boards of Appeal of the EPO (Boards) have held that a technical invention which uses a computer program is patentable. It follows the European legal tradition and the legal history of the EPC that patents can

---

[95] Art 52(2)(C).

only be granted for inventions which have a technical character.[96]

The requirement of a technical contribution can be regarded as the major difference between the EPO and the US. In the US an invention is simply required to be within the technological arts and the invention using a computer or software becomes a part of the technological arts if it provides a useful, concrete and tangible result.

[2] Recent Cases

Vicom case[97]

The *Vicom* case has authority on what "*computer program as such*" means and what constitutes a "*mathematical method*". The patent application related to a method and apparatus for digital image processing which involved a mathematical calculation on numbers representing points of an image. Algorithms were used for smoothing or sharpening the contrast between neighbouring data elements in the array. The Board of Appeal held that a computer utilizing a program to carry out a technical process is not a claim to a computer program *as such*.

IBM cases[98]

The invention in *T 0935/97-3.5.1* related to detecting where a second window in a computer display overlies part of the first window, obscuring information in a portion of the first window. The invention allows the obscured information in the first window to be displayed in another portion of the first window not obscured by the second window.

---

[96] Commission of the European Communities, *The Patentability of Computer-implemented Inventions*, Brussels, 19 October 2000. This documents is available on the Commission's DG Internal Market website at http://europa.eu.int/comm/internal_market/en/intprop/indprop/index.htm.

[97] T 208/84, 15 July 1986 Computer-related invention/Vicom (OJ 1987, 14).

[98] These are *T 0935/97-3.5.1* [1999] R.P.C. 861 and *T 1173/97-3.5.1* 1999 OJ EPO [609].

On the other hand, the invention in *T 1173/97-3.5.1* related to resource recovery in a computer system and a capability to continue an application while the system resynchronises a failed or incomplete procedure.

In both cases, the Examination Division accepted both system and method claims, but rejected computer program product claims. However, regarding the computer program claims, the Boards noted the following:

- Programs for computers must be considered as patentable inventions when they have a technical character.
- A technical character can be found in further technical effects deriving from the execution of the instructions given by a computer program. Such a further technical effect could have the necessary technical character where it causes the software to solve a technical problem.
- The central question is what *"further technical effect"* can lead to this subject-matter being patentable. If a computer program product produces such a further technical effect when run on a computer, such a program product can produce such a further technical effect.

These two decisions nullified, and reversed, a passage in the Guidelines for the Examination in the EPO (EPO Guidelines),[99] where it is stated that a computer program claimed by itself, or as a record on a carrier is not patentable irrespective of its contents. Under the reasoning of the Boards, a computer program claimed by itself is not excluded from patentability if the program brings about a technical effect which goes beyond the normal physical interactions between the program and the computer. Furthermore, the Boards of Appeal reached the important conclusion that it does not

---

[99] *Guidelines for the Examination in the EPO*. Chapter IV Patentability of Part C-Guidelines for Substantive Examination. http://www.european-patent-office.org/legal/gui_lines/e/index.htm accessed 7 April 2001.

make any difference whether a computer program is claimed by itself or as a record on a carrier. In the past, the EPO had refused to grant claims to a computer program on a computer-readable medium and a carrier on the ground it amounted to "computer program as such". In these two decisions, however, a technical effect can be derived either from the execution by the computer of the instructions provided by the software program, or from the solution of a technical problem enabled by the software.

These decisions offered significant new patent protection for software by recognizing not only recording media but also the program itself. The EPO has produced a "practice note on the patentability of programs for computers". It also indicates that a technical effect is caused when programs are run on a computer and such a computer program might be claimed by itself or as a program product or as a record on a carrier.[100] The *Directive on the Patentability 2002/0047 (COD)* draws the conclusion that all programs when run in a computer are technical because a computer is a machine and thus, have a technical character.[101]

In these two decisions, the EPO has gone further than the *Beauregard* court in the US, by suggesting that patents may be obtained to a computer program product without reference to some sort of carrier. This would give more flexibility in determining patent protection for software-implemented inventions.[102] After these cases the EPO came to treat computer program product claims in much the same way as they are treated in the US and Japan. The *Directive on the Patentability 2002/0047 (COD)* stipulates that a computer-implemented invention[103] may be claimed as (1) a product, (i.e. as a

---

[100] John Adams, Puay Tang and Daniel, *Patent protection of computer programmes*, ECSC-EC-EAEC Brussels-Luxembourg, (University of Sussex at Brighton) 2001, (afterwards, Adams and Tang).

[101] *Directive on the Patentability 2002/0047 (COD)*, p. 7. The *Directive* cites the *Pension Benefits System* case (*T 931/95* OJ EPO 10/2001, Board of Appeal of the EPO, decision of 8 September 2000) to draw the conclusion.

[102] Robert Hart, *The Economic Impact of Patentability of Computer Programs*.

[103] The *Directive on the Patentability 2002/0047 (COD)* defines "computer-implemented inventions" as

programmed computer, a programmed computer network or other programmed apparatus), or as (2) a process carried out by such a computer, computer network or apparatus through the execution of software (Article 5). This suggests that a software-related invention can be claimed as (1) a programmed computer and/or computer network, or as (2) a process carried out by such a computer or computer network.

[3] The Current Situations in the EPO[104]

The current legal situation in the EPO on patent protection for computer-implemented inventions is unsatisfactory because of its uncertainty. According to the EPC, computer programs as such are excluded from patentability. According to the EPO Guidelines for Examination in the European Patent Office (EPO Guidelines), a computer program claimed by itself and a computer program stored on a disk or other carrier are excluded from the patent protection. However, about 13,000 patents using computer programs have been granted by the EPO.[105] 75% of these patents are held by non-European companies. Most small and medium enterprises (SMEs) in the software industry are not aware that patent protection can be obtained for this type of inventions.[106] This uncertain situation has negatively influenced software development.

On the other hand, considering the scope and the impact of harmonization, the

---

to mean 'any invention the performance of which involves the use of a computer, computer network or other programmable apparatus and having one or more *prima facie* novel features which are realized wholly or partly by means of a computer program or computer programs.' See the *Directive on the Patentability 2002/0047 (COD)*, Article 2.

[104] Commission of the European Communities, *Consultation Paper on "The patentability of computer-implemented inventions"*, Brussels, 19 October 2000.

[105] More than 20,000 of patents for computer-implemented inventions have been granted by the EPO. See *Directive on the Patentability 2002/0047 (COD)*. There are about 13,000 European patents covering software. See *Promoting Innovation Through Patents*, Communication from the Commission to the Council, the European Parliament and the Economic and Social Committee, 5 February 1999, pp.12-13.

[106] See Adams and Tang.

European Commission[107] is striking a balance between promoting innovation and ensuring adequate competition. In accomplishing a harmonized approach to the patentability of computer-implemented inventions in the European Community, one of the key elements is the "technical effect" requirement. A computer program causes a technical effect when run on a computer.[108] This technical effect may consist in the control of an industrial process or the working of machines by the computer. It may also exist in the internal functioning of the computer itself or its interface (e.g. user interface for a business management system). The technical effect may already be known in the prior art.[109]

To involve an inventive step, a computer-implemented invention must make a technical contribution to the art. The further requirement that the invention must make a technical contribution is assessed under the inventive step criterion. It is an important limitation on patentability that the technical contribution also has to be non-obvious to a person skilled in the art. A technical contribution is the difference between the technical

---

[107] The European Commission derives its constitutional functions from the Treaty of Rome. The Commission can promulgate Directives to harmonize the national laws applied by EU member states in their national Patent Offices. The European Commission operates at the very heart of the European Union. The Commission has used its right of initiative to transform the framework provided by the treaties establishing the European Communities into today's integrated structures. Although the Commission makes the proposals, all the major decisions on important legislation are taken by the ministers of the Member States in the Council of the European Union, in co-decision (or, in some cases, consultation) with the European Parliament. The Commission acts as the EU executive body and guardian of the Treaties. It represents the common interest and embodies the personality of the Union. The 20 members of the Commission are drawn from the 15 EU countries. One of the principal tasks here is to secure the free movement of goods, services, capital and persons throughout the territory of the Union. Because of its right of initiative, the Commission is charged with making proposals for all new legislation. Once a Commission proposal has been submitted to the Council of Ministers and the European Parliament, the three institutions work together to produce a satisfactory result. In agreement with the Commission, the Council can amend a proposal by a qualified majority (if the Commission does not agree, the change requires unanimity). The European Parliament shares the power of co-decision with the Council in most areas and has to be consulted in others. When revising its proposals the Commission is required to take Parliament's amendments into consideration. See http://europa.eu.int/comm/role_en.htm accessed 8 June 2002. The EPO, however, is not an institution of the European Community. With regard to the EPO, see this study, Chapter 1. Introduction, p. 1.

[108] *T 0935/97* of 4 February 1999, [1999] R.P.C. 861 and *T 1173/97* of 1 July 1998, 1999 OJ EPO [609].

[109] Ibid.

features of the invention and the technological state of the art. It can exist in the problem solved by the invention, or in the means constituting the solution of the underlying problem, or in the effects achieved in the solution of the problem. The technical contribution may constitute an alternative solution for an already solved technical problem or for an already known technical effect.[110]

In the light of the studies on the impact of patents for computer-implemented inventions on innovation and competition as well as European business, the European Commission believes that protection for computer-implemented inventions should be harmonized without any sudden changes in the legal position, and particularly any extension of patentability to computer programs "as such".[111] The Commission believes that for the time being, the Community should not extend the patent protection for computer-implemented inventions by doing without the technical contribution requirement.

3.4 Software Patents in Japan

[1] Statutory Patentability in Japan

The definition of 'an invention' in Japan under Article 2(1) of the Japanese patent law is broad. It is defined as *"the highly advanced creation of technical ideas utilizing natural laws"*. This requirement means that an invention must be technical and also must use the laws of nature. The law of nature contrasts with laws of man, such as rules of chess. The requirement in Japan that an invention must be technical ideas is similar to the

---

[110] Ibid. See also the Consultation Paper.
[111] *Directive on the Patentability 2002/0047 (COD).*

technical effect criterion in the EPO, in that the requirement of 'utilizing a law of nature' can be compared with that of 'technical effect' in the EPO.[112] In addition, if 'hardware resources' in the list of utilizing laws of nature under the Implementing Guidelines (1997)[113] are interpreted as a computer, it is found that the requirement of 'utilizing a law of nature' is similar to that of the EPO. In the EPO, 'technical effect' may consist in the following:[114] (1) the control of an industrial process or the working of a piece of machinery by the computer; (2) the internal functioning of the computer itself or interfaces under the influence of the program (e.g. user interface for a business management system), and (3) a computer program running on a computer.[115]

There have been many debates regarding the "technical idea" requirement because the scope of technology, especially in relation to computer programs, is uncertain and changing continuously. The debate was mainly over the issue of whether an algorithm was patentable from the perspective of utilization of the laws of nature. The JPO has created examination guidelines for such inventions. Presently, the JPO provides specific standards concerning utilization of the laws of nature in "2.2.1 Basic Concept" and "2.2.2 Examples of Means Utilizing Natural Laws" of "Chapter 1. Computer Software Related Inventions"[116] in the *Implementing Guidelines for Inventions in Specific Fields* (1997). The guidelines were derived from the conventional interpretation of Article 2 of the Japanese Patent Law.

---

[112] http://europa.eu.int/comm/internal_market/en/intprop/indprop/studyintro.htm accessed 4 May 2001.

[113] See this study, 4.5 [3] The Current Situation in Japan. There are three categories in order to be considered to utilize a law of nature;
(1) Control for hardware resources, or processing for controlling
(2) Information processing based on the physical or technical properties of an object
(3) Information processing in which hardware resources are used

[114] *The Patentability of Computer-implemented Inventions*, Commission of the European Communities, 19 October 2000.

[115] The *Sohei* case *T 769/92*.

[116] www.jpo-miti.go.jp/ accessed 15 December 2001.

[2] Implementing Guidelines for Inventions in Specific Fields[117]

[a] Basic Concept of Utilizing the Laws of Nature

The Implementing Guidelines provide specific standards for software inventions as to the laws of nature in the JPO. According to the Guidelines, an examiner must identify the problem to be solved and its solution. If the solution is judged to fall under any of the following three categories, the invention is considered to utilize a law of nature: (1) control for hardware resources, or processing with respect to the control; (2) information processing based on the physical or technical properties of an object; (3) information processing in which hardware resources are used.

When the solution is a mathematical algorithm, a law of nature itself, a natural phenomenon, a mathematical expression of a law of nature or a natural phenomenon, or when it is related solely to human science, the invention is not utilizing a law of nature. Also, even though the solution is considered to utilize a law of nature, when it is no more than a "mere recording of a program or data on a storage medium", the claimed invention is considered as non-statutory. This standard is based on Article 2 of the Japanese Patent Law.

When a software invention is expressed in a sequence of processes or operations, the invention can be defined as a process invention. When a software invention is expressed as one or more functions performed by the invention, the invention can be regarded as a product invention by specifying the functions. "A computer-readable storage medium having a program (or structured data) recorded thereon" can be stated

---

[117] JPO, *Implementing Guidelines for Inventions in Specific Fields* (1997). http://www.jpo.go.jp/infoe/sisine.htm accessed 15 December 2001 (afterwards, *Implementing Guidelines (1997)*).

in a claim as a product invention.[118] These storage mediums may constitute statutory invention, because when the structure of recorded data identifies how a computer processes the data (i.e. when the functional interrelationships exist between the structure and the process), the processing is utilizing natural laws.

The term "computer-readable storage medium having a program recorded thereon" is interpreted as a computer-readable storage medium which is particularly suitable for installing or executing the program. Therefore, when judging whether a material is included in the "storage medium" or not, it should be considered whether the material is suitable for the above purpose.

[b] Examples of the Solution Utilizing Natural Laws

Control for hardware resources, or processing with respect to the control

'An apparatus, a method and a storage medium (containing a computer program recorded thereon) for controlling rate of fuel injection for an automobile engine' are regarded as controlling for hardware resources, or processing regarding the control.

The problem to be solved by the invention is to improve the combustion efficiency and the output power of the engine during the stages of hard acceleration or deceleration. This invention intends to achieve the optimum fuel/air mixture ratio by controlling fuel injection rate in line with changing conditions to improve the combustion efficiency and the power output of the engine.

The solution to the problem is processing for control in which the transition of the rate of engine revolutions is detected in addition to the rate of engine revolutions, and the optimum rate of fuel injection is determined by using the values of the rate of engine

---

[118] See also the *Beauregard* and the *IBM cases*.

revolutions and the transition of the rate of engine revolutions. As the processing is the control for hardware resources and is based on the physical or technical properties of an object, the solution is regarded as utilizing law of natures.

Information processing based on the physical or technical properties of an object

For example, 'an image processing method by computer and a computer-readable storage medium containing a computer program for image processing recorded thereon' is regarded as a case of information processing based on the physical or technical properties of an object.

This invention relates to a picture quality improvement method in image processing by a computer. An object of the invention is to provide an image processing method which can achieve a required compensation sufficiently and easily. The problem to be solved by the claimed invention is regarding to the handling of the data as physical quantities obtained by picking up the image with an optical reading means, and the issue of how to achieve a required and sufficient correction to the physical quantities of pixel which has a blur depending on the characteristics of the optical reading means. The solution to solve the problem is "processing in which each pixel picked up is multiplied by a parameter of digital filter B (a high-pass filter) which realizes inverse characteristics of the optical reading means".

The processing is based on the physical or technical properties of an object, since the picture quality is improved by using filter parameters which have inverse characteristics of producing the blurring. Thus, the solution is considered as utilizing the laws of nature.

Information processing in which hardware resources are used

'Information processing in which hardware resources are used' includes the following cases: (1) computer-readable record medium that stores student performance management data; (2) a game machine; and (3) an apparatus for predicting daily sales of commodities. ('(3) An apparatus for predicting daily sales of commodities' will be discussed in the section of 3.5 BMPs in Japan.)

(1) A computer-readable record medium that stores student performance management data

The invention relates to the techniques for the management of performance by high-school students, where teaching staff establish mark books listed by student names and subjects. Increasing numbers of students and subjects result in an increased volume of performance data, leading to omissions and errors. Under these circumstances, computers have been used to check student performance subject by subject.

If conventional performance management techniques were to be computerized as they are, the result might involve the duplicate entries for the same students and the same subjects. This would reduce the operating efficiency of the computer memory and increase the search time required. The problem to be solved by the invention is to provide an efficient method of managing students' subject-based performance data using a computer. The solution to the problem is "processing for access to performance data sorted according to subjects and stored in the performance files using a subject-based pointer stored in the student file".[119]

As this processing can be identified by the specific data structure based on the matters described in the claim, it is a means utilizing the laws of nature. In addition,

---

[119] *Implementing Guidelines (1997).*

since some matter[120] defining the invention in the claim suggest how the hardware resources of the computer are utilized in the processing, the solution to the problem utilizing the laws of nature can be regarded as something more than "mere processing of information by using a computer".

(2) Game machine

As for the game machine, there can be two claims as follows:[121]

[Claim 1]

A computerized card game machine, comprising:
means for assigning specific points of a score to a set of cards dealt, according to the complexity of the hand involved.

[Claim 2]

A computerized card game machine, comprising:
means for memorizing a data table for a scoring hand (i.e. a hand of cards dealt that scores points) in which a given set of cards is matched to specific scoring hand data, and a score data table in which the scoring hand data are matched to the score data;
means for assigning corresponding scoring hand data by retrieving said scoring hand data table based on a set of cards selected, assigning corresponding score data by retrieving the score data table on the basis of the applicable scoring hand data, and outputting all of the scoring hand data and total points scored.

The invention relates to computerized card game machines. Usual computerized card game machines extract a hand of cards dealt, score the points from among a given set of five cards dealt by the computer, determine the scores based simply on the number of scoring hands, and display the results obtained. The usual practice of scoring the same points for any types of hands reduces the amusement of the game.

The problem to be solved by the invention is to provide a card game machine that

---

[120] Ibid. E.g. "means for assigning corresponding scoring hand written data by retrieving said scoring hand written data table based on a set of cards selected, assigning corresponding score data by retrieving the score data table on the basis of the applicable scoring hand written data, and outputting all of the assigned scoring hand written data and total points scored".
[121] Ibid.

makes the game more exciting by assigning different point scores to a set of cards depending on the complexity of the hand involved. The solution to the problem is "processing for computing scores using computer hardware resources assigning different scores to a set of cards dealt according to the complexity of the hand involved".[122]

Since claim 1 has no matter which suggests how the hardware resources of the computer are utilized in the processing the scores earned, the processing falls under "mere processing of information by using a computer". However, in terms of claim 2, since some matters[123] defining the invention in the claim suggest how the hardware resources of the computer is utilized in the processing, the solution to the problem is something more than "mere processing of information by using a computer" and the invention is regarded as a statutory invention.

[c] Information Processing by Use of a Computer[124]

Even if the claimed invention is carrying out information processing by use of a computer, if the claim has no matters which suggest how the hardware resources of the computer are utilized in the processing, then the processing falls under "mere processing of information by using a computer" which does not constitute a statutory invention. For example, in terms of "an apparatus for calculating the sum of natural numbers from 'n' to 'n+k' by using a computer", the claim (1) is written as follows:[125]

> *An apparatus for calculating the sum of natural numbers from 'n' to 'n+k' by using a computer, comprising:*

---

[122] Ibid.
[123] Ibid. See supra n.120.
[124] Ibid., pp. 24-29.
[125] Ibid.

means for inputting natural numbers 'n' and 'n+k';
arithmetic means for obtaining the sum 's' of natural numbers from 'n' to 'n+k' with s = (k+1)(2n+k)/2; and
means for outputting the calculated result.

The problem to be solved by the invention in the claim (1) is to calculate the sum of natural numbers from 'n' to 'n+k' by using a computer in a shorter time than conventional way. The solution to the problem is "processing by use of computer, where the sum is calculated with s = (k+1)(2n+k)/2" that is mathematical formula. Thus, even though the solution utilizing the laws of nature is "processing by use of computer hardware resources", the processing falls under "mere processing of information by using a computer", since the claim (1) has no matters that suggest how the hardware resources of the computer is utilized in the processing.

However, the invention can be considered as a statutory invention, if the claim (2) is written as follows:[126]

An apparatus for calculating the sum of natural numbers from 'n' to 'n+k' by using a computer, comprising:
means for inputting natural numbers 'n' and 'n+k';
'n' storage means for storing input 'n';
'n+k' storage means for storing input 'n+k';
means for calculating 'k' by reading 'n' from 'n' storage means and 'n+k' from 'n+k' storage means respectively;
'k' storage means for storing said 'k';
arithmetic means for calculating the sum 's' of natural numbers from 'n' to 'n+k' with s = (k+1)(2n+k)/2, where 'n' and 'k' being stored in said 'n' storage means and 'k' storage means respectively; and
means for outputting the calculated result.

The problem to be solved by the claimed invention is to calculate the sum of natural numbers from 'n' to 'n+k' from entered natural numbers 'n' and 'n+k' by using a computer in a shorter time than conventional way. The solution to the problem of the invention of claim (2) is:[127]

---

[126] Ibid.
[127] Ibid.

processing in series consists of:
processing for calculating 'k' by reading 'n' from 'n' storage means and 'n+k' from 'n+k' storage means respectively;
processing for storing said 'k' in 'k' storage means; and
processing for calculating the sum 's' of natural numbers from 'n' to 'n+k' with s = (k+1)(2n+k)/2, where 'n' and 'k' being stored in said 'n' storage means and 'k' storage means respectively.

Since the matters defining the invention in the claim (2) suggest directly how the hardware resources of the computer are utilized in the processing, the solution utilizing the laws of nature is something more than "mere processing of information by using a computer". Thus, the invention of the claim (2) is considered as a statutory invention.

On the other hand, in terms of "a process for calculating the sum of natural numbers from 'n' to 'n+k' by using a computer", the claim (3) can be written as follows:[128]

A process for calculating the sum 's' of natural numbers from 'n' to 'n+k' by using a computer with s = (k+1)(2n+k)/2.

Even though the solution utilizing natural laws is "processing by using a computer", since the claim (3) has no matters suggesting how the hardware resources of the computer are utilized in the processing for calculating, the processing is considered as "mere processing of information by using a computer".

[d] Inventive Step (Non-obviousness)[129]

Deciding whether an inventive step exists involves judging whether or not the claimed invention could have been easily made by a person skilled in the art. Therefore, it is important to understand the ordinary creative ability of a person skilled in the art at the time of filing. The following are considered as exercises of an ordinary creative ability

---

[128] Ibid.
[129] Ibid., pp. 9-13.

of a person skilled in the art: (1) mere selection of an optimal material among the known materials; (2) determination of an optimal numerical value range; (3) replacement of means with a well-known equivalent; and (4) modification of a design in the specific application of technical matters. If these are the only differences between the invention and the cited invention, it is considered that the invention would have been easily made by a person skilled in the art.

Application to other fields

Since procedures or means used in a computer applied invention are often common in functions or operations, regardless of the field the invention belongs to, it is natural to expect that a person skilled in the art would have tried to apply a procedure or means of the software-related invention of one field to another field in order to realize the same function and operation.

For example, if a "medical recovery system" exists as a cited invention, applying the means used in the known "medical recovery system" to a "commodity retrieval system", where the function or operation is shared in common by both systems, can be regarded as mere 'an application to other field' having no inventive step.

Supplement or replacement by a commonly known means for systematization

Because software-related inventions are usually realized in a system integrating hardware and software, it is natural that a person skilled in the art would try either to supplement a commonly known means for systematization as a constituent element of the system, or to replace part of constituent elements of the system with a well-known equivalent.

Implementation by software of function which are otherwise performed by hardware

It is expected that a person skilled in the art would try to implement functions performed by hardware by means of software having common functions.

Systematization of human transactions

Systematization of existing human transactions by a computer is regarded as an exercise of ordinary creative ability of a person skilled in the art, if the transactions can be realized by a routine application of usual system analysis and system design technologies.

Recording a program or data on a computer-readable storage medium

Even if a limitation of "recording a program or data on a computer-readable storage medium" is added to a claim, the inventive step is denied.

[3] Patent Protection for Software-related Inventions in Japan[130]

The 1993 Examination Guidelines state that both "computer programs" and "recording media recorded with computer program" are not categorized as inventions. The 1997 Guidelines acknowledge subject-matter of "computer programs" and "recording media recorded with computer programs" in certain cases. The guidelines admit a claim only for "recording media recorded with programs" on the basis of the requirements for description, insisting that "recording media recorded with programs" are inventions of

---

[130] www.jpo.go.jp/tousie/pdf/bukai_report_e.pdf accessed 10 June 2002, *Report Presented by the Intellectual Property Committee of the Industrial Structure Council*, December 2001

products, but that the category of "programs in themselves" is not definite. The 2000 revision of the Patent Examination Guidelines states that "computer programs" can be described in claims as inventions of products regardless of whether or not they are recorded on recording media.

CHAPTER 4

Business Method Patents and Protection by Patent Law

4.1 Introduction

E-commerce has been affecting, and has been affected, by IP. Rapid growth of e-commerce has introduced BMPs. Methods of doing business using the Internet were regarded as a process. In this respect, BMPs involve a process which is a kind of algorithm. In the US, the test for the patentability of a business method is whether its subject-matter has practical utility that provides a useful, concrete and tangible result. In the EPO, the patentability of a business method depends on whether its subject-matter has a technical character. In Japan, it is whether its subject-matter is a technological idea using the laws of nature.

[1] Historical Background to BMPs

There are two key elements in the rapid growth of BMPs. One is information technology development and the other is the pro-patent policy initiated by the US.

[a] Information Technology Development

The rapid growth of the Internet and e-commerce is regarded as one of the main reasons of the new recognition of BMPs.[1] The innovation of information technology

---

[1] Richard Poynder, *Patenting Software*, UK Department of Trade & Industry, 2001.

development has brought a significant increase in Internet-related patent applications.[2]

The digital revolution is changing the world fundamentally. It has brought about the new digital economy represented by innovative business methods. At the moment there are roughly 150-200 million persons around the globe connected to the Internet.[3] The advent of the Internet has significantly changed ways of conducting business. The Internet has enabled sales channels to be easily set up and business ideas to be directly incorporated into business. In the US, the incorporation of ideas into business methods through the Internet can acquire patent protection. Examples of such cases are the Attention Brokerage patent (US patent 5,794,210) of the Cyber Gold Inc., and "Name Your Price" (US patent 5,794,207) of the Priceline. The transition of business practices from the non-Internet world to the Internet has made the invention novel. In response, software developers, financial services firms, on-line businesses, traditional hardware manufacturers, and even agricultural companies have applied to protect their business methods by patents.

[b] The Pro-patent Policy in the US

The pro-patent policy is one of the main reasons for the recognition of BMPs and the rapid growth of them. From the early 1980, the US economy started to suffer a severe crisis caused by the decline in industrial competitiveness. To revitalize industry, the government introduced the "pro-patent policy".[4] The pro-patent policy was characterized as being "strong protection" and "wide-ranging protection". This policy

---

[2] See USPTO Press Release, March 1999 and Wired News, 4 May 1999.

[3] See "eGlobal Report," at *http://www.emarketer.com* (reporting there were 130.6 million active users as of 1999); and Nua Internet Surveys (*http://www.nua.ie*) (171 million as of May 1999). IDC (International Data Corporation) also reports that there are 497.7 million Internet users worldwide (as of 2001). http://www.nic.or.kr/news/news_20020115.pdf accessed 14 February 2002.

[4] www.jpo-mini.go.jp/index 17 February 2001 accessed.

gave strength to the patents and expanded the scope of patent protection for software. In 1982 the US Supreme Court officially recognized computer software as patentable.[5] Aiming at the clarification of software patents, in 1987 the USPTO reformed its examination guidelines. In 1996 to recognize legally the patenting of recoding media for recording programs, examination guidelines were reformed.

In the early stages of the pro-patent policy, there were a number of cases where Japanese manufactures paid huge amounts for patent infringements. These pro-patent trends have spread to Japan and Europe; Japan officially adopted a pro-patent policy in 1997[6] and the EPO recognized patents for recording media in the *IBM* cases.[7]

[c] The Pro-patent Policy in Japan[8]

Since a report of the "Council for Consideration of Intellectual Property Rights in the 21st Century" was issued (Japan, April 1997), pro-patent policies have been gradually and firmly established in Japan. Recognizing the gap between the US and Japan in technical capabilities,[9] Japan has started to pursue a pro-patent policy. In the technology trade, the US was remarkably in the black, while Japan was in the red in the 1986-1995 periods.[10] Statistical data indicate that Japan was running a deficit of 32.9 billion dollars (accumulated deficit from 1987 to 1996), and that the US was gaining a

---

[5] See the *Diehr* case.
[6] Council for Consideration of Intellectual Property Rights in the 21st Century, Japan, April 1997.
[7] See *T 0935/97-3.5.1* [1999] R.P.C. 861 and *T 1173/97-3.5.1* OJ 1999 EPO [609].
[8] JPO, *Pro-patent Era in Japan*, February 1997. And see also the Planning Subcommittee of the Industrial Property Council, *Report of the Planning Subcommittee of the Industrial Property Council-To the better understanding of pro-patent policy*, November 1998. http://www.jpo-miti.go.jp/indexj.htm accessed 13 March 2001.
[9] Japan's patents are those of improvement, and there are few basic patents. See JPO, *Annual Report on Patent Administration* (1998).
[10] "Monthly Report on International Trade Balance' by Bank of Japan, "Survey of Current Business" by US Department of Commerce.

tremendous surplus of 147.1 billion dollars (accumulated surplus from 1987 to 1996).[11]

The patenting of recording media for recording programs was legally recognized in the 1997 operation guide for examination of software-related inventions. The concept of the doctrine of equivalents was adopted in the Japanese Supreme Court in February 1998.[12] The Tokyo District Court made a decision recognizing damages in the amount of approximately 3 billion yen, the highest amount ever, in a legal suit over patent infringement in October 1998. Under the circumstances, Japan came to adopt their own pro-patent policy, and this policy resulted in a flourish of BMPs in Japan.

[2] Classification of BMPs

It is very difficult to classify all the kinds of BMPs because new ideas in business methods are continuously coming out as technology develops. BMPs can be classified by the parties involved in the transactions: business to business; business to consumer; and consumer to consumer.[13] The JPO classifies as follows:[14]

[a] BMPs regarding e-commerce
  (1) Transaction Brokerage
  (2) Billing
[b] BMPs regarding finance

BMPs can be classified by the contents of the inventions, e.g. Internet shopping, Internet brokerage business, Internet marketing, E-cash, supply chain, online billing and

---

[11] JPO, *Towards the International Harmonization of Intellectual Property Rights Systems in the 21st Century*, www.jpo-miti.go.jp/tousie/chapter1.htm accessed 20 June 2000.
[12] The *Spline Shaft* case, judgment of 24 February 1998, Japan Supreme Court, 1630 Hanji 35 (1998).
[13] Henry Koda, *Business Model Patent*, Nikei Kogyo Shinbunsha, (Tokyo, Japan), 2000.
[14] Kazuo Makino, Sidney H.Weeks and Kanji Kawamura, *Business Method Patents*, Nihon Keijai Shinbun, 7 June 2000.

online gambling. For the purpose of this study, they can be classified as follows:[15]

A. *Known* BMIs which have technical application and are implemented by a *known* computer automation system.

B. *New* BMIs which have technical application and are implemented by a *known* computer automation system.

C. *Known* BMIs which have technical application and are implemented by a *new* computer automation system.

D. *New* BMIs which have technical application and are implemented by a *new* computer automation system.

E. *New* business methods implemented by a computer but *without* technical application.

F. *New* business methods *without* the use of computer.

Technical application means practical use of technology, and it may include a technical problem and its technical solution. According to the recent EPO cases, a computer program causes a technical effect when run on a computer.[16]

In terms of the Trilateral Study, business methods of Choice A are regarded as non-patentable. In the US, however, Choice A and even F could be patentable.[17]

Business methods of Choice B have a technical basis but the innovatory step is not technical in nature. These business methods in Choice B might be regarded as being patentable because a claim should be considered as a whole. One of the reasons is that distinguishing the claims in Choice B and C in determining whether the claims have inventive step entail dissecting the claim into technical and non-technical parts, and then

---

[15] Korean Industrial Property Office (KIPO), *Examination Guidelines for E-commerce Related Inventions*, August 2000. The author reorganized the classification underlying in the Guidelines. During the reorganization, the author transformed 'industrial application' into 'technical application' and added '*New* BMIs implemented by *known* automation system'.

[16] See *T 0935/97-3.5.1* and *T 1173/97-3.5.1*.

[17] In the US business methods without using software or hardware can be patented, if useful, concrete and tangible results are provided. E.g. *In re Fox* (176 USPQ 340) and *In re Warmerdom* (31 USPQ 2d 1754).

ignoring the non-technical part. The other is that newly developed business methods usually require new or, at least, reorganized computer software. German jurisprudence did not exclude the possibility that business methods having a technical aspect could be patentable even if the only contribution that the invention makes is non-technical.[18] Under UK jurisprudence, however, software-related invention that amounts to a method of doing business is considered unpatentable even if a technical contribution may be found.[19] Considering the fact that the great majority of the responses to the Consultation Document of 01. 11. 2000 by the UK Patent Office (UKPO) opposed patents for computer-implemented business methods if there is no technological innovation,[20] Choice B is likely to be opposed by European companies or individuals.[21] According to the Explanatory Memorandum of the *Directive on the Patentability 2002/0047 (COD)*, if there is no technical contribution, (i.e. if the contribution to the state art exists wholly in non-technical aspects, e.g. a method of doing business) the invention cannot be regarded as patentable.

Choice C and D raise little question about their being patentable subject-matter, because they have technical application as well as new computer automation system. In Korea, business methods of Choice E might be regarded as non-patentable because they do not have any concrete means for industrial application. In terms of Choice F, there was a consensus in the UK that patents for business methods where no computer is involved should not be granted.[22] In Korea, business methods of Choice F are regarded

---

[18] See *Directive on the Patentability 2002/0047 (COD)*, p. 10; "Automatic Sales Control" case [1999] GRUR 1078; and "Speech Analysis Apparatus" [2000] GRUR 930.
[19] See *Merrill Lynch* [1989] RPC 569.
[20] www.patent.gov.uk/about/consultations/annexa.htm accessed 4 May 2001. (*Should Patents be Granted for Computer Software or Ways of Doing Business?*)
[21] This might be because the answerers did not understand appropriately how patent claims should be interpreted.
[22] www.patent.gov.uk/about/consultations/annexa.htm.

as non-patentable.[23]

4.2 Critical Issues in BMPs

In the EPO, business methods had been excluded from patent protection until the *IBM* cases. In the US, before the recent two decisions, the *SSB*[24] and the *AT&T*,[25] the patentability of methods of doing business was limited by two long-standing principles, the "business method exception" and the "mathematical algorithm exception".

The business method exception assumed that methods of doing business did not fall within any class of patentable subject-matter. The USPTO had used the business method exception for many years to reject inventions based on methods of doing business. However, in its 1996 Guidelines, the USPTO stated that statutory subject-matter should include computer-related inventions that involve business methods. The two recent decisions referred to above completely eliminated the business method exception. The mathematical algorithm exception also limited the patentability of methods of doing business on the Internet, because business methods on the Internet involve computer programs that also involve mathematical algorithms. The exception can be traced back to earlier the US Supreme Court decisions that categorized "laws of nature, natural phenomena, and abstract ideas" as unpatentable subject-matter. In the recent decisions, however, the CAFC drastically limited the exception.

The non-obviousness requirement seems to create difficult problems for the patentability of Internet business methods, especially when an invention involves an

---

[23] KIPO, *Examination Guidelines for E-commerce Related Inventions*, pp. 13-15.
[24] 149 F.3d (Fed. Cir. 1998).
[25] *AT&T Corp. v. Excel Communications, Inc.*, 172 F.3d 1352 (Fed. Cir.), cert. Denied, 120 S. Ct. 368 (1999).

Internet business method that was transferred from non-Internet business methods. It is necessary to examine whether the transformation from the real world to the Internet is obvious or not. Judging the patents that the USPTO has issued, such as an "Internet-based shopping cart", the conversion of a non-Internet business method to an Internet-based was considered as a non-obvious improvement. The USPTO has granted Internet BMPs, just for transferring business practices from the real world to the Internet. Although pure business methods themselves are not subject to patent protection, business methods using computers and the Internet are regarded as being subject to patent control.[26] However, inventions restructured to the specific structure of the Internet, would be more likely to be considered non-obvious than inventions transferred simply as above. Unexpected improvements, commercial success, long felt but unsolved needs, and failure of others can be used as evidence of non-obviousness.

4.3 BMPs in the US

[1] Introduction

Even though there is no official definition of BMPs, they are classified in the US Patent Classification as 705 Class. Business methods without using software or hardware can be patentable if they have a useful, concrete and tangible result.[27] The USPTO interprets the term "useful, concrete and tangible result" as "immediately applicable and valuable in real world". The patentability of business methods involving mental steps

---

[26] It was stated by the Tripartite Director-Generals Meeting in June 2000.
[27] E.g. *In re Fox* (176 USPQ 340) and *In re Warmerdom* (31 USPQ 2d 1754).

cannot be denied merely because mental steps are involved.[28]

[2] Case Law Developments

In this section, cases concerning business methods will be examined in chronological order. The focus will be centred on the *Hotel Security,*[29] the *Merril Lynch*[30] and the *SSB*.

Before the Hotel Security cases

The *Hotel Security* case is often cited as the first case covering the patentability of business methods in the US. However, other cases can be found before the *Hotel Security*. *United States Credit System Co. v. American Indemnity Co.* (*US Credit System*)[31] can be considered as the very first case.

The *US Credit System* invention was a method to prevent excess losses from bad debts. Its distinctive features were to create charts on the average percentage of losses for each business field of the insured companies to evaluate the risks concerned, to conclude insurance contracts, to cover only the losses that exceeded the average percentage, and to establish limits to be covered and the degree of risks. The sheets used in this method had headings and description spaces for items, including the name of the insured company, the name of the underwriter, and the business field. Thus, the contents of the invention were operations generally conducted in accounting or book-keeping.

---

[28] Intellectual Property Institute (Japan), *Report on Protection Method of New Field (Business Methods)*, March 2000, p. 263.
[29] *Hotel Security Checking Co. v. Lorraine Co.*, 160 F. 467 (2d Cir. 1908).
[30] *Paine, Webber, Jackson & Curtis, Inc. v. Merril Lynch, Pierce, Fenner & Smith, Inc.* 564 F.Supp. 1358 (D.Delaware, 1983).
[31] 51 F.751 (C.C.N.C. Illinois; 1892); 53 F.818 (C.C.S.D. New York; 1893); 59 F.139 (2nd Cir. 1893).

The court stated that, since the invention was related to a general method of business transactions, the invention was not statutory subject-matter.

On the other hand, in *Thompson v. Citizens' National Bank of Fargo* (*Thompson*, 1892)[32] and the *Benjamin Menu Card Co. v. Rand McNally & Co. et al* (*Benjamin*, 1894),[33] we can find a positive attitude to inventions involving a business idea. In the *Thompson* it was held that an improved account book could constitute statutory subject-matter, because it was a new and useful improvement. This opinion was also supported in the *Benjamin* case. The *Benjamin* invention consisted of conventionally used bills and menus to prevent waiters and cooks from conspiring.

The Hotel Security case

The issue of the case was validity of the patent right for an invention entitled "A Method and Means of Cash Registering and Account Checking." The purpose of the invention was to prevent illicit acts resulting from conspiracy between customers and waiters by accurately confirming the waiter in charge and the amount entered into the cash register in hotels or restaurants.

The court noted that it was necessary to examine first whether the invention fell under the category of arts, and then whether it was new and useful. With respect to the first point, the court stated that a system of transacting business disconnected from the means of carrying out the system is not art. However, the court did not make clear whether the invention claimed in the patent was a "system of transacting business which is disconnected from the means for carrying out the system."

With regard to the second point, the court judged that a sheet of paper and a paper

---

[32] 53 F.250 (8th Cir. 1892).
[33] 210 F.285 (C.C.N.D. Illinois; 1894)

slip, which were the physical means to work the invention, had little new and useful distinctive features. The court concluded that the art described in the specification to work the invention existed from the past and was not novel.

Even though the case was not decided on the subject-matter ground, the USPTO interpreted it in an excessively broad manner. The USPTO began to reject all claims to business methods. This practice was eventually codified in the USPTO's Manual of Patent Examining Procedure (MPEP) §706.03. The relevant provision stipulates that though seemingly within the category of process or method, a method of doing business can be rejected as being non-statutory.[34] This provision remained in the MPEP until 1996. Because of the provision, few business method patent applications had been filed during the period between the *Hotel Security* (1908) and 1996.[35] Moreover, those involving these were applications tried to disguise the true nature of their claims.

The business method exception was established on the US case law[36] and the USPTO practice.[37] However, because the *Hotel Security* was finally decided not on subject-matter but on novelty, the court's comments concerning the subject-matter of business methods were not binding. This led to much confusion in subsequent cases concerning BMPs.

From the Hotel Security case to the Merril Lynch case[38]

After the *Hotel Security*, we can find both positive and negative decisions for business related inventions. The following are positive decisions: *Rand, McNally & Co. v.*

---

[34] MPEP §706.03(a) (August 1993).
[35] Brenda Sandburg, *Patent Applications Flow Freely*, Legal Times, 22 February 1999, at 12.
[36] E.g. *In re Wait*, 24 USPQ. 88 (1934). *In re Howard and Brobeck*, 394 F.2d 869, "*I would place the affirmance of the board's decision upon the ground that the application discloses merely a method of doing business and is therefore for an unpatentable invention.*"
[37] MPEP § 706.03(a) (August 1993).
[38] 564 F.Supp. 1358 (D.Delaware, 1983).

Exchange Scrip-Book Co. (*Rand*, 1911); *Cincinnati Traction Co. v. Pope* (*Cincinnati*, 1913); *In re Johnston* (1974); and *In re Deutsch* (1977). The following are negative decisions: *In re Sterling* (1934); *In re Wait* (1934); *In re Patton* (1942); and *In re Maucorps* (1979).

In the *Rand* case (1911), the court indicated that the concept of the invention could not be regarded as a mere business method. The court noted that the invention was not for a method, but for a ticket that was a physical object, and that it was impossible to conduct the method without such an object. The court stated that equipment could often become the first part of a new business method by being newly designed and provided for use, and that patent rights had been granted to such equipment. The *Cincinnati* court (1913) also held that a ticket could constitute statutory subject-matter, because the ticket was used in a business method and was categorized as a manufacture.

On the other hand, in *In re Sterling* (1934), the court judged that the invention of a bank check was unpatentable. The court stated that the physical structure of the bank check was the same as part of existing checks, and that, although the items described on the check were different, such descriptions had been excluded categorically from statutory subject-matter as being printed matter. The court pointed out that, in the *Cincinnati* case, patentability of the invention was judged based not on what was described on the ticket, but on the physical structure of the ticket. In addition, the court stated that, if the bank check had presented some new and useful physical form, the situation would have been different.

In sum, in these three cases we can find that patentability had been recognized to new and useful physical objects used in business methods.

The invention in *In re Patton* (1942) was a fire protection system to counter

attacks on airplanes. The court clarified that, even if new and useful, a system for conducting business did not constitute statutory subject-matter if it was disconnected from the means for putting the system into practical use or working the system. This interpretation was also applied in *Loew's Drive-In Theater, Inc. v. Park-In Theater, Inc.* (1949).

The inventions from the late 1960s cases were those worked using computers. The invention in *In re Johnston* (1974), a record-keeping system for financial information utilizing digital computers, was interpreted as being a machine or a system involving a machine. In *In re Deutsch* (1977) the court noted that the method of an automatic control system for a manufacturing plant could not be considered as a business method just because the application included a step to convert business data into mathematical language readable by computers controlling the manufacture plant.

In *In re Maucorps* case (1979), however, we find a negative attitude. The court considered whether it fell under an algorithm, and held it to be non-statutory subject-matter. The invention was a computer system for optimizing sales organizations and sales activities, which aimed at calculating the optimum number of times a salesperson should visit a customer within a certain period.

The Merril Lynch case[39]

The *Merril Lynch* invention was entitled "*Securities Brokerage-Cash Management System*", which was related to the data processing methodology and apparatus for effectuating the cash management account program (CMA).

Paine Webber contended that the patent was invalid because it did not claim a

---

[39] *Paine, Webber, Jackson & Curtis, Inc. v. Merril Lynch, Pierce, Fenner & Smith, Inc.* 564 F.Supp. 1358 (D.Delaware, 1983).

"process, machine, manufacture or composition of matter". It argued that the claims defined only the combination of familiar business systems in which a margin brokerage account, money market funds and checking/charge account have been connected to exchange financial information. It contended that business methods and systems could not form a statutory subject-matter. It also argued that in an attempt to obscure the fact that the invention was business system, the claims were drafted to recite a combination of various "means" for performing certain functions. Paine Webber urged the court to focus on the product of the patent, that is, the services provided by the CMA to customers, rather than on the method that the CMA operates.[40]

The court clarified the fact that the focus should be not on the product of a computer program, but on the operations of the program on the computer. The court concluded that the patent included statutory subject-matter because the claims represent a method of operation on a computer to effectuate a business activity.[41]

Historical decisions in recent cases

Finally, the *SSB* case declared that the business method exception no longer exists. This decision caught the attention of industries which had paid no attention to patents before such as banking, security business and education business. Despite of almost 90 year presumption that business methods were not proper patent subject-matter unless embodied in some tangible form, such a presumption has not been formally endorsed. In the *SSB* case, however, this exception was virtually abolished, and BMIs (having a computer as a requisite element) came to be recognized as inventions constituting statutory subject-matter. The *AT&T* reaffirmed this decision.

---

[40] Robert P.Merges, *Patent Law*, n.p., 1997, p. 145.
[41] Ibid., pp. 145-146.

[3] The Current Situation in the US

After the *SSB*, the USPTO and the CAFC have been criticized on the ground that mundane or obvious business methods are being recognized as patentable. To enhance the quality of the examination of BMIs, various efforts are being made. The USPTO issued the Business Method Patent Initiative: An Action Plan.[42] It also prepared "A USPTO White Paper".[43] In Congress, the US Rep. Berman and Rep. Boucher proposed the Business Method Patent Improvement Act of 2000,[44] which required publishing all business method patent applications after 18 months, creating an administrative opposition process which is less costly than litigation, and lowering the burden of proof to predominance of the evidence for anyone challenging a business method patent.

4.4 BMPs in the EPO

[1] Introduction

Mathematical methods, methods of doing business and programs for computers as such, are excluded from patentability by the EPC. This exclusion was an established practice until *T 1173/97 Computer Program Product* (*IBM*). The EPO[45] regards the standards presented in the *SSB* case, that the invention shows a "useful, concrete and tangible

---

[42] www.uspto.gov/web/offices/com/sol/actionplan.html Accessed on 18 October 2000.
[43] *A USPTO White Paper: Automated Financial or Management Data Processing Methods (Business Methods)*, USPTO, 19 July 2000.
[44] US Rep. Berman and Rep. Boucher introduced a bill in the House 3 October 2000 to enhance the quality of BMPs.
[45] The EPO was set up by the contracting states to the EPC. See this study, Chapter 1.

result", as not satisfactory. After the *IBM* case, the EPO is considering the amendments of the current examination guidelines. The EPC is also being examined for revision. The elimination of the exclusion of "computer programs" from the EPC has been agreed. It is suggested that it must be in harmony with TRIPs 27.[46] Others suggest that inventions should involve technical solutions. Current practice in the EPO is that inventions must meet the requirement of technical character. BMPs issued in the EPO would appropriately be categorized as those wherein a technical character has been ascribed to a software based invention relating to a business scheme.[47] However, elimination of a "method of doing business" from the EPC 52(2)(c) is not being considered.

[2] Patents in the EPO[48]

The following are European patents which would fall into the BMPs category:

EP 0,195,098 (*Avedas Inc.*): "System for reproducing information in material objects at a point of sale location". This is regarded as the EP equivalent of the E-Data patent.[49]

EP 0,209,907 (*Sohei*): "Computer system for plural types of independent management and method for operating a general-purpose computer management system."

EP 0,407,026 (*Reuters Ltd.*): "Distributed system and method for matching buyers

---

[46] Article 27(1) of the TRIPs Agreement reads, "Patents shall be available for any inventions, whether products or processes, in all fields of technology, provided that they are new, involve inventive step and are susceptible of industrial application". http://www.wto.org/english/tratop_e/trips_e/t_agm0_e.htm accessed 1 June 2002. Non-compliance by WTO member countries may lead to trade sanctions.

[47] Erwin J. Basinski, *Business Method Patents in Europe: A Saussurean Explanation*, World E-commerce & IP Report, Volume 1, Issue 7, April 2001 (afterwards, Basinski, *Business Method Patents in Europe*), p. 12.

[48] Ibid.

[49] US 4,528,643, "System for reproducing information in material objects at a point of sale location".

and sellers". This is related to foreign currency trading system.

· EP 0,762,304 (*Citibank AG.*): "Computer system for data management and method for operating said system."

EP 0,784,276 (*Sun Microsystems Inc.*): "Shopping cart for the web."

EP 0,836,727 (*American Express*): "Methods and apparatus for providing a prepaid remote entry customer account."

T 1002/92 (1996) EPOR1 (*Petterson*): "A computer controlled queuing equipment for directing people in a single queue to one of a multiplicity of servers."

[3] Recent Important Cases

The EPO rejects those inventions that do not provide a technical contribution. BMIs may not be patentable in the EPO unless the invention provides a technical contribution outside of a method of doing business. An example of the approach is given in the official headnote to the decision in the *IBM/Card Reader*.[50] The invention of the *IBM/Card Reader* involved the use of an electronic application form to determine whether the user was entitled to access to, for example, a cash dispensing machine. The court noted that parts of the method claimed were merely instructions for using the machine and a claim which was essentially a business operation, did not have a technical character. The court held that the claim was a method for doing business as such and therefore unpatentable.

Recently the European Technical Board of Appeals of the EPO (Technical Board) rendered three important decisions on the patentability of BMIs: the *Queuing*

---

[50] *T 854/90* [1993] OJ EPO 669.

system/Petterson,[51] the *Sohei*[52] and the *Pension benefits system.*[53] In the *Queuing system/Petterson* case, a system for determining the queue sequence for serving customers at plural service points was held to be patentable. The Technical Board held that the problem to be solved was the means of interaction of the components of the system, and that since this was a technical problem, its solution was patentable. It appears that claiming intangibles is possible in the EPO, provided that such intangibles are closely linked to a technical system.

The *Sohei* case opened a way for a business method to be patentable. The patent was a computer system for plural types of independent management including financial and inventory management, and a method for operating the said system. The court held that it was patentable because 'technical considerations were applied' and 'technical problems were solved'. Thus, the Technical Board considered the invention to be patentable, although it was dealing with a method of doing business.

The *Pension* case,[54] however, denied the patentability of an "Improved Pension Benefit System", for which the USPTO had granted a patent. The appeal was directed against the decision of the examination division of the EPO which refused European patent application No. 88 302 239.4 on the ground that the invention related to a method for doing business. The claimed inventions include "a method for controlling a pension benefits program by administering at least one subscriber employer account" and "an apparatus for controlling a pension benefits system". The applicants argued as follows:[55]

a) The data processing and computing means according to the invention formed

---

[51] *T 1002/92* [1995] OJ EPO 605.
[52] *T 769/92* [1995] OJ EPO 525.
[53] *T 0931/95-3.5.1*, OJ EPO 10/2001 [441] Boards of Appeal of the EPO, decision of 8 September 2000.
[54] Ibid. The case concludes that all programs when run in a computer are technical because a computer is a machine. See *Directive on the Patentability 2002/0047 (COD).*
[55] Basinski, *Business method patents in Europe*.

the technical basis for implementing a new pension system.

b) The present invention consisted of a technical tool serving an actuary and therefore was different from "doing business" in terms of Article 52(2)(c) EPC.

c) The claims were directed to the processing of data which were related to physical entities, as was the case in decision *T 208/84 Computer-related invention/Vicom* (OJ 1987,14) and thus not directed to a pension system *as such*.

d) Relying on the "technical character" of invention was not justified, since such a criterion was not set up by the EPC as a requirement for patentability. The reliance on technical requirements for such applications was outmoded, and according to the *Sohei* case and the *Queuing system/Petterson* case,[56] the practice of the EPO had changed and opened the era of business methods to patent protection.

e) The invention had a technical character, since it applied to the apparatus claims and to the method claims which comprised the use of data processing means.

The Technical Board responded as follows:

a) Having technical character is an implicit requirement of the EPC for an invention to be patentable.

b) With regard to method claims, the question is whether the method of claim 1 represents a method of doing business as such. All the features of the claim are the steps of processing and producing information which have purely administrative, actuarial and/or financial character. Since processing and

---

[56] *Queuing system/Petterson T 1002/92* [1995] OJ EPO 605.

producing such information are typical steps of business and economic methods, the invention is merely a method of doing business as such, lacking any technical character, and thus it is excluded from patentability under Article 52(2) and (3) EPC.

c) There are no arguments or facts in the specification which indicate that the steps of the method solve any technical problem or achieve any technical effect.

The Board concluded that methods involving only economic concepts and practices of doing business are not inventions within the meaning of Article 52(1) EPC, and that a feature of a method which concerns the use of technical means for purely non-technical purpose (and/or for processing purely non-technical information) does not necessarily confer a technical character to such a method. The Board distinguished this invention from that of the *Sohei*[57] on the ground that the method claim in the *Sohei* specified the functional features of the computer system which required technical considerations. The Board also distinguished the invention of the *Petterson* case as claiming an apparatus, not a method.[58]

Regarding the apparatus claim, the Board made the interesting observation that the specific wording of Article 52(2) EPC referred to "schemes, rules and methods" as being excluded from patentability, but did not mention an "apparatus" as being excluded. The Board concluded that an apparatus constituting a physical entity or concrete product suitable for performing or supporting an economic activity is an invention within the meaning of Article 52(1) EPC, but the claimed subject-matter does not involve an inventive step, since the invention has no technical problem or contribution

---

[57] The *Sohei T 769/92* (OJ 1995, 525).
[58] *T 0931/95* (Reasons, para. 4).

to the prior art and the improvement by the invention is essentially economic one, which can not contribute to inventive step.[59]

The Board also made a distinction between technical effect and technical contribution as follows: The "technical contribution" relates to what is claimed, whereas the "technical effect" relates to the character of the subject-matter. Even though "technical contribution" plays no role in determining whether the apparatus claim is patentable, it is required to assess inventive step requirement.

The *Pension* case gives us the following suggestions as to what must be done to acquire BMPs in the EPO:[60]

a) Make sure to characterize the invention as having a technical effect. To do this, describe the technical problem and the solution. Describe the way the invention solves the technical problem in a concrete and useful way. The technical solution, which becomes the "technical contribution" to the prior art, describes the novelty and inventive step.

b) Remember that claiming the subject-matter as involving computers, computer networks, or other conventional programmable apparatus gives a technical effect advantage only where it is necessary to show a "further technical effect". A further technical effect may consist of processing data which represents physical entities, and makes the computer system more efficient, faster, more secure; however, whichever of these is claimed, you must include some limitation in the claim which implements or articulates this effect.

c) Include an apparatus claim to avoid being regarded as business method as such.

d) In the preamble of the claim, avoid writing "a method for doing some sort of

---

[59] Ibid. (Reasons, para. 5-7).
[60] Basinski, *Business method patents in Europe*, p. 15.

non-technical thing (e.g. financial data processing)" because it is difficult to convince the examiners that it is not a business method as such.

[4] The Current Situation in the EPO

European policy makers think that the European innovation system has not been as successful in commercialising inventions as that of America or Japan. They believe that changes to patent law could encourage European firms to patent more, and help to foster and maintain competitiveness against their American or Japanese competitors. However, extended patent protection will increase the cost for small users because of high licence fees. Even though there has been some uncertainty on the patentability of software-related inventions, BMPs have also been granted in the EPO.[61] Because the EPO has already started to grant patents on business methods, the current issue for European policy makers is whether they allow more BMPs to foster the development of e-commerce, or they continue allowing fewer BMPs.[62]

However, some European companies, especially SMEs have not been aware of the possibility of patenting software and business methods. Rapid action was required to clear the patentability of software and BMIs, and to prevent European companies being dominated by Japan or the US, where little restriction on patenting software-related inventions existed.[63] The consultation on the Green Paper showed the need to harmonize in the light of the recent developments in the US and to clarify the current legal situation. However, the main consequences for SMEs should be assessed.[64]

---

[61] Robert Hart, *The Economic Impact of Patentability of Computer Program*. Eg, EP 0 572 403 B1 (March 1995); EP 0 762 304 B1 (March 1997).
[62] Adams and Tang.
[63] The Commission's Consultation Paper, p. 12.
[64] The study was conducted by the Intellectual Property Institute, London, on behalf of the Commission

According to the Consultation Paper[65] issued by the European Commission,[66] a computer-implemented invention that makes a technical contribution and not merely a contribution in the business field will be patentable even if its application concerns methods for doing business or mental acts. The presence of such non-technical features (business application) will not preclude a finding of a technical contribution. "The technical contribution shall be assessed by consideration of the difference between the scope of the patent claim considered as a whole, elements of which may comprise both technical and non-technical features, and the state of the art."[67] When assessing inventive step, technical and non-technical features should be assessed without discrimination. Thus, a BMI may be patentable if a non-obvious technical contribution is present. However, if the technical contribution to the state of the art lies merely in non-technical features, the invention will not be considered as involving an inventive step.[68] The *Directive on the Patentability 2002/0047 (COD)* ensures that patents for pure business methods will not be granted because they do not meet the requirement for technical contribution.

The EU Committee favourably responded to the conclusions of the Commission's Consultation Paper.[69] The Committee stated that the current European requirement for "technological contribution" has prevented (and should prevent) the granting of BMPs

---

and finalized in March 2000.

[65] Commission of the European Communities, *Consultation Paper on "The patentability of computer-implemented inventions"*, Brussels, 19 October 2000. This documents is available on the Commission's DG Internal Market website at http://europa.eu.int/comm/internal_market/en/intprop/indprop/index.htm.

[66] While the EPO was established by the EPC, The European Commission derives its constitutional functions from the Treaty of Rome. The Commission can disseminate Directives to harmonize the national laws applied by EU member states. Because of its right of initiative, the Commission is charged with making proposals for all new legislation.

[67] *Directive on the Patentability 2002/0047 (COD)*, Article 4.

[68] Ibid., p. 14.

[69] The EU Committee, *The EU Committee's Response to the Commission's Consultation Paper on Patents for Computer Implemented Inventions Initial Discussion*, 12 January 2001. (This paper is available on-line at the EU Committee website: http://www.eucommittee.be.)

similar to the most controversial patents granted in the US. The Committee also noted that many of the proposed legislation's recommendations for changes in the US patent system reflected existing European practice, and that they indicated that the current European practice was already well suited to deal with BMPs.[70]

The UKPO published the results of its consultation on the patentability of software and business methods on 12 March 2001.[71] The consultation's conclusions indicate that the UKPO's current position on patentability is that (1) software should be patentable if it is for a "technological innovation"; (2) business methods as such should not be patentable; and (3) clarification of unclear part of law is necessary.

In short, the UKPO supports the status quo with further clarification.

[5] Summary

The EPO case law is continuously developing independently of the political process considering the advantages and disadvantages of BMPs.[72] According to the latest case (the *Pension* case),[73] to be patentable, a BMI must have technical character and the innovation must concern technical subject-matter.

Since it is difficult to reverse existing EPO case law by restricting the availability of patents in this sector, the question in future will be whether there should be a more

---

[70] *The EU Committee's Response to the Commission's Consultation Paper on Patents for Computer Implemented Inventions.*

[71] www.patent.gov.uk/about/consultations/conclusions.htm accessed 11 May 2001.

[72] Jürgen Siepmann, a German attorney specializing in software law, explains the relations between the European institutions and the risks which the European Commission is taking by legislating on the software patent question: "The EPO is not an institution of the European Community. Its Technical Boards of Appeal are only inferior chambers within an intergovernmental administrative body... By proposing to remove all limits on patentability, the EPO created an unexpected backlash of public opinion... With its Greenpaper and its Consultation Paper, the European Commission has gradually adopted the position of the EPO." http://www.eurolinux.org/news/warn01C/index.en.html accessed 7 June 2002.

[73] *T 931/95-3.5.1*, OJ EPO 10/2001 [441] issued on 8 September 2000.

liberalization of the patentability of software or BMIs.[74] According to the *Directive on the Patentability 2002/0047 (COD)*, the European Commission concludes that it is right that for the time being at least, the Community should not extend the patent protection for computer-implemented inventions by doing without the technical contribution requirement.[75] The *Directive* makes it clear that 'a computer-implemented invention in which the contribution to the prior art does not have a technical character should be considered to lack inventive step *even if the (non-technical) contribution to the prior art is not obvious.*'[76] In determining the technical contribution, the invention must be assessed as a whole without discrimination between technical and non-technical features. Thus, an invention that is related to a method of doing business may be patentable if there is a non-obvious technical contribution. However, if there is no technical contribution, i.e. if the contribution exists wholly in non-technical aspects (e.g. purely a method of doing business), there will be no patentable subject-matter.

4.5 BMPs in Japan

[1] Statutory Patentability

According to the definition of a statutory invention which is provided by the Japanese Patent Law,[77] it is required to satisfy three requirements: (1) utilization of a law of nature, (2) technical idea and (3) creativeness. 'Technical idea' is interpreted as a

---

[74] Ari Laakkonen, *European and UK Software and Business Method Patents are in a Holding Pattern*, World E-commerce & IP Report, Volume 1, April 2001, p. 20.
[75] *Proposal for a Directive of the European Parliament and of the Council on the Patentability of Computer-implemented Inventions*, 2002, p. 11.
[76] *Directive on the Patentability 2002/0047 (COD)*, p. 14.
[77] See this study, 3.4 [1] [a] Basic Concept of Utilizing the Law of Nature.

specific means for accomplishing a certain purpose, which is workable and repeatable.

The most contentious requirement is utilization of a law of nature. A mere mental activity, a pure theoretical law or an artificial agreement, is not considered to utilize a law of nature. Therefore, it is possible to interpret BMIs as non-statutory inventions, since their composition is supposedly based on the knowledge of social science or artificial arrangements.[78]

The Japanese system has exceptions which are designated in "Implementing Guidelines for Industrially Applicable Inventions" announced concurrently along with "Implementing Guidelines for Inventions in Specific Fields".[79] An invention which corresponds to any of the following cases does not conform to a statutory invention:

(1) Natural laws as such

(2) Mere discoveries

(3) Inventions contrary to natural laws

(4) Laws other than natural laws, and inventions in which solely such laws are utilized

(5) Personal skills (which are acquired through personal experience and can not be shared with others as a knowledge due to lack of objectivity)

(6) Mere presentations of information (where feature resides solely in the content of the information, and the main object is to present information)

(7) Aesthetic creations

(8) Those for which it is clearly impossible to solve the problem to be solved by any means presented in the claim.

It is contentious whether business methods may come under the above case (4), and

---

[78] Ryuta Hirashima, *Changes in Subject-matter under the US Patent Law*, p. 26.
[79] *Implementing Guidelines (1997).*

whether utilization of a law of nature is found in business methods.

[2] Cases in Japan

With respect to BMIs, there are several cases. In 1953, an invention of a method to encipher a telegram into a single alphabet was denied as a non-statutory subject-matter because it did not use any apparatus, nor applied a means that utilized a law of nature.[80] In November 1953, an invention comprising a telegram into a single Japanese letter was also denied as a non-statutory invention for not utilizing a law of nature.[81] In 1956 an invention of an advertising method using electric poles was ruled as non-statutory for the same reason.[82]

Compared with the inventions mentioned above, most of recent BMIs use computers and are implemented with software. These characteristics make it appropriate to regard these inventions as software-related inventions specifically applied to business methods.

Regarding BMPs, there are protest cases. One is AT&T's "Karmarkar" patent. The patent was published in 1996 and protested on the ground that a mathematical solution cannot be patented. Since the JPO rejected the protest, the protest parties appealed to the court in December 1999. Another is the Citi Bank's patent which is related to 'Electronic Money System'. The patent was published in November 1995 and protested by many Japanese banks. The JPO decided that the Citi Bank's patent was obvious (24 December 1997). The Citi Bank appealed to the Trial Board (13 April

---

[80] Supreme Court decision of 30 April 1953, Civil Litigation Precedents Vol. 7 No. 4, p. 461.
[81] Tokyo High Court decision of 14 November 1953, Administrative Litigation Precedents Vol. 4 No. 11, p. 2716.
[82] Tokyo High Court decision of 25 December 1956, Administrative Litigation Precedents Vol. 7 No. 12, p. 3157.

1998). The Trial Board held that the JPO's decision should be revoked (17 November 1999).[83]

[3] The Current Situation in Japan

Under the Implementing Guidelines (1997), to be considered to utilize a law of nature, an invention should fall under any of the following three categories:

(1) Control for hardware resources, or processing for controlling

(2) Information processing based on the physical or technical properties of an object

(3) Information processing in which hardware resources are used

BMIs are not likely to fall under category (2). But they could fall category (1) if the inventions include control of hardware resources or processing in their elements. As defined above, since BMIs require hardware as a platform to operate software, they would fall under category (3), as long as the business method can be provided by utilizing a computer system. Meanwhile, the inventions are unlikely to be a "mere recording of a program or data on a storage medium". The distinction between a "mere processing of information by using a computer" and category (3) remains problematic. However, the guidelines indicate that, if the claim has no matters that suggest how the hardware resources of the computer are utilized in the processing, the processing falls under a "mere processing of information using a computer". If the invention utilizes hardware, the invention will be interpreted as information processing in which hardware resources are used. In short, the distinction appears to be a mere difference in the form of the claim. Most BMIs would be considered to utilize a law of nature under the

---

[83] Shinpanpyeong 10-5570 Ho. See http://www.ipdl.jpo.go.jp/Shinpan/spsogodbk.ipdl.

Implementing Guidelines (1997).

As a specific example of business methods which can be affirmed as a statutory invention, the Implementing Guidelines (1997) exemplify: "An apparatus for predicting daily sales of commodities". This invention relates to a system for predicting daily sales necessary for ordering commodities based on data on fluctuation factors in past sales including such as the weather, a day of the week, events, status of competing stores, etc. In the Implementing Guidelines (1997), this apparatus is regarded as a statutory invention, because the hardware resources of the computer are utilized in the processing. In addition, it is explained that, since the claim has some matters which directly suggest how the hardware resources of the computer are utilized in the processing, the processing is something more than "mere processing of information by using a computer".

Realizing that BMPs can have a great influence on various industries, the Examination Standards Office of the JPO published the '*Examination of Business-related Inventions*' (December 1999).[84] This document shows that BMIs have been examined according to the guidelines, and that they will be positively examined. The JPO also announced the "*Policies Concerning Business Method Patents*" (December, 2000), in which the JPO established following policies:[85]

(1) Clarification of examination standards by revision of examination standards

(2) Collection of more wide-ranging data for business methods

(3) User-friendly search system for prior art

(4) Utilization of experts and development of examiners/appeal examiners to handle applications for BMPs

---

[84] www.jpo-miti.go.jp/ accessed 13 March 2001.

[85] For more detailed information, see the revised examination standards (draft) on the JPO website.

(5) Efforts aimed at compatible practices among the Trilateral Offices

(6) Information promulgation/explanation of BMPs by releasing BMPs on its website

In sum, under the Japanese Patent Law BMIs are not categorically excluded from statutory inventions, if they are considered to be utilizing a law of nature.[86]

4.6 Summary of Software Patenting[87]

All three jurisdictions permit claims to computer programs on a carrier. In all three jurisdiction business methods are patentable. The fundamental difference between the US and Europe is that in Europe to be patentable computer-implemented BMIs should be of a technical character,[88] while in the US computer program related inventions are of the technological arts and they need only provide useful, concrete and tangible results. In the US, the restrictions on patenting business methods are negligible.

While the US does not have statutory exclusions for inventions, the EPC has exclusions which include programs for computers and methods of doing business. In practice, however, a number of patents have been granted on computer programs and methods of doing business. The scope of software-related inventions that can be recognized to have technical character has been expanded by decisions made by the EPO.[89] "Technical contribution" appears to be more restrictive than "useful, concrete

---

[86] Ryuta Hirashima, *Changes in Subject-matter under the US Patent Law.*

[87] Robert Hart, *The Economic Impact of Patentability of Computer Programs.*

[88] Ibid.

[89] Since 1990 statutory subject-matter of an invention has been determined on the basis of whether or not the invention makes a 'technical contribution' to prior art. However, the *T 769/92* on the *Sohei* case in 1995 adopted 'technical consideration' as an examination standard. The *T 1173/97* on the IBM case in 1998 confirmed that a computer program is statutory subject-matter because a computer program that has "further technical effects" has technical features. The judgement also ruled that a computer program itself as well as a computer program as a record written on a medium is patentable subject-matter. See Report

and tangible result". This difference has resulted in fewer bad patents being granted in Eufope than in the US.[90] However, inconsistency between the statutory patentability and case law in the EPO causes uncertainty which is bad for business.[91] If a proper action is not taken rapidly, many software patents and BMPs (which otherwise would have been obtained by the European companies) would be occupied by the US companies.

In sum, practices in patenting software as well as statutory patentability of computer-related inventions are different between patent jurisdictions. Inconsistency is also found in each jurisdiction. There is an apparent discord between the statutory patentability in the EPC and the practice of the EPO. The *IBM* cases reversed the EPO Guidelines.[92] In the US, the subject-matter patentability of software inventions involving mathematical algorithm was denied if failed in the FWA test.[93] However, in the *SSB* and the *AT&T*, the CAFC declared that the FWA test has little applicability to determining the presence of statutory subject-matter. Until the *SSB* court declared that the business method exception no longer exists, the patentability of business methods had long been denied unless embodied in some tangible form. However, methods of doing business (even without using software or hardware) can now be patentable.

In this state of affairs, before considering whether following the US practice in

---

Presented by the Intellectual Property Committee of the Industrial Structure Council. Available at www.jpo.go.jp/tousie/pdf/bukai_report_e.pdf accessed 10 June 2002. With respect to the criterion of 'technical contribution' and 'technical consideration', see Final Report by PbT Consultants (under contract number PRS/2000/A0-7002/E/98), *The Results of the European Commission Consultation Exercise on The Patentability of Computer Implemented Inventions.* P. 21. http://europa.eu.int/comm/internal_market/en/indprop/comp/softanalyse.pdf accessed 10 June 2002.

[90] However, the European system may be considered as broader than that of the US in that claims for computer programs not on a carrier are acceptable.

[91] The existence of such uncertainty and divergences in legal protection can have a negative effect on investment decisions. See *Directive of on the Patentability 2002/0047 (COD).*

[92] After the cases, a computer program claimed by itself is not excluded from patentability.

[93] Even in applying the FWA test, the CAFC had been inconsistent. The court applied the test to *In re Shrader* and *In re Warmerdam* to reject the subject-matter patentability of the inventions. However, the CAFC did not apply the test to *In re Alappat* and *In re Lowry*.

patenting software, or persuading the US to follow the EPO practice, or coordinating with the US and Japan to harmonize the patentability of software and business methods, it is reasonable to examine whether or not the existing legal regimes are appropriate for protection of software innovations. This is because the main problems in software patenting (i.e. impeding follow-on innovation, establishing entry barrier) might be originated from the *very* nature of the existing legal regimes. If the main problems are resulted from the lack of harmony between the characteristics of software innovations and the existing legal regimes, it is necessary to consider providing alternative protection. This is because only trying to revise the existing regimes cannot solve the fundamental problems of the existing regimes which are based on exclusive property rights. If an alternative proposal is recognized by the US and Japan[94] (and the other countries in future) as more effective and appropriate than existing regimes for solving the fundamental problems of promoting software innovation without impeding follow-on innovation in sequential innovation, it could be introduced in these three jurisdictions first and at international level in future. With this common approach, the problems of disharmony between jurisdictions could be solved more easily than with the existing regimes.

---

[94] These three jurisdictions are responsible for the great majority of patent applications in the world.

CHAPTER 5

Economic Review of Software Protection by Existing Regimes

5.1 In General

The primary goal of the patent system is to encourage technological innovation and the transfer and dissemination of technology.[1] The patent system can be regarded as a means to induce the disclosure of secrets in return for the grant of exclusive rights, and as a scheme for promoting inventions.[2] It also can be seen as a contract between an inventor and the public. Upon voluntary request by an inventor, the public grants patent rights to the inventor and the inventor discloses the invention to the public.[3] The public can use patent information to improve the technology used in the patent, or to develop new applications. On the other hand, through the patent, inventors and their financial backers are able to protect their investments.[4]

According to classical economic theory, patenting stimulates innovation. Through a guaranteed monopoly, the patent allows innovative companies to reap the benefits of their investments in R&D by setting high prices or licensing. Patenting, however, tends to weaken competition, increase consumer prices, make market operation more inflexible and entail considerable management costs. The patent system may also constrain innovation if the protection it gives is too broad.[5]

Software is a valuable artefact whose know-how is largely evident in distributed

---

[1] Article 7 of the TRIPs Agreement.
[2] Ove Granstrand, *The Economics and Management of Intellectual Property*, 1999, Edward Elgar Publishing Limited, p. 31.
[3] Ibid., pp. 71, 82.
[4] In this respect, the scope of a patent may be determined in proportion to the amount of the contribution through the information the inventor has disclosed to the public.
[5] Cohen & Lemley, *Patent Scope*. Some of the early biotechnological patents underline this point.

products.[6] This makes the products vulnerable to rapid, inexpensive copying that undercuts the initial developer's opportunity to benefit from the value he has contributed to the market, thereby undermining his incentive to invest in software development. There are several reasons why the know-how to create software is so accessible: (1) software products need little specialized mass-production know-how; (2) they are rich in design know-how; and (3) they are more susceptible in some respects to reverse engineering than many traditional industrial products.[7]

Software is easy, simple and inexpensive to mass-produce. Software, especially interactive software, is rich in design know-how from which much of the value of software arises. Many of the hard-won innovations embodied in design are apparently displayed in operation.[8] Other know-how embodied in software resides in the detail of its internal construction, such as algorithms, data structures and control structures. The innovation in program internals often lies in constructing new ways to organize and structure these information components. One way to access the internal design is to decompile (reverse engineer). As the technology for the reverse engineering of programs improves, the internal know-how of programs may come nearer to the surface of the product.

In sum, software is vulnerable to easy acquisition of equivalence, the product of which is indistinguishable from the original to users, and therefore, can be a market substitute for the original. The crucial concern is that it is quick and easy to copy a software innovation that was very expensive to develop. If the cost of copying is small enough, when compared with the cost of innovation, it can destroy the lead-time.[9] Since

---

[6] *A Manifesto*, Section 2.
[7] Though not pharmaceuticals.
[8] *A Manifesto*, Section 2.
[9] Lead-time is recognized by businessmen as one of the best ways of protecting innovation. See www.business.auc.dk/druid/conferences/nw/paper1/kingston.pdf accessed 10 July 2001.

lead-time is important for the innovator's opportunity to recoup its investment, reducing lead-time can destroy incentives to invest in software innovation. The recent appearance of the Internet makes software distribution almost free and instantaneous, resulting in the loss of lead-time, which generally exists in physical product industries due to the need to obtain the raw materials, to organize manufacturing facilities, and to distribute the manufactured products.

Existing legal regimes (i.e. patent, copyright and trade secrecy) have been used to try to provide appropriate protection for know-how embodied in software products. Although trade secrecy law has a long history of protecting industrial compilations of applied know-how, it cannot protect behaviour embodied in software products because such know-how cannot be kept a secret, and because trade secrecy law has long regarded reverse engineering of products in the market as a fair means of acquiring trade secrets.[10] An experienced programmer can run a program to study its component behaviours and can often learn everything necessary to make a functionally indistinguishable program.[11] Nonetheless, trade secrecy may be the principal way that software developers today protect algorithms.[12] When an algorithm is embodied in program code, its know-how is not borne on the surface of the product, although it may be discovered by decompilation. Protecting an algorithm by trade secrecy will not necessarily afford its developer a permanent advantage in the market. Other developers may independently come up with the same algorithm or may be able to infer the algorithm by analysing the embodying program's behaviour. The idea can even be

---

[10] See, e.g., *Kewanee Oil Co. v. Bicron Corp.*, 416 U.S. 470, 476 (1974) (construing Ohio statutes and Restatement of Torts 757). If the defendant takes the market-destructive approach of appropriating the information from material given to it in confidence by the plaintiff, rather than actually doing the reverse engineering, courts sometimes find trade secret misappropriation.

[11] Though he cannot usually reverse engineer source codes. See *A Manifesto*, Section 2.

[12] Ibid., Section 5.2, n.302. Code, ideas, and concepts may be treated as trade secrets so long as they are not obtainable through other products by lawful means, including reverse engineering.

patented by others if they develop it independently. A trade secret is lost when the secret becomes general knowledge. In the meantime, however, trade secrecy protection for algorithms will tend to give developers lead-time approximately covering the expense and risk of creating their innovations. If an algorithm is obvious, other developers are likely to figure it out relatively quickly and reimplement it in their own code if the algorithm is superior to what they previously used. The less obvious an algorithm is, the longer it will likely take competitors to figure it out, and the greater will be the natural lead-time advantage to the innovator. Either way, the innovator is likely to have some natural lead-time in the market.

Recently, especially in the US, patent protection for software has been extended on the basis of the increased necessity to protect innovation in the software industry. However, there have been many debates between those for and against extending the patentability of software creations. Advocates assert that expanded protection will provide stronger incentives to develop new technologies, and that expanded protection will make it possible for new companies to secure finance. The authors of the ETAN (European Technology Assessment Network) Working Paper on the *Strategic Dimensions of Intellectual Property Rights in the Context of S&T Policy* conclude that IPRs are vital for innovation, and can foster innovation if approached in the right way.[13] According to the European Commission study, *Innovation Policy in a Knowledge-Based Economy*,[14] IPRs are crucial to information-based industries such as pharmaceuticals,[15]

---

www.library.carleton.edu/staff/michael/acarticle.html accessed 7 January 2002.

[13] ETAN Expert Working Group, *Strategic dimensions of Intellectual Property Rights in the context of S&T Policy,* Report prepared for the European Commission, June 1999, p. X, cited by Adams and Tang, p. 92, n.4. This is available at http://europa.eu.int/comm/research/era/pdf/ipr-expertgroupreport.pdf 10 June 2002.

[14] European Commission, *Innovation Policy in a Knowledge-Based Economy*, p. 58.

[15] See European Commission, Anthony Arundel Merit, *Patents – the Viagra of innovation policy?* Internal report to the Expert Group, Prepared as part of the project "*Innovation Policy in a Knowledge-Based Economy*" commissioned by the European Commission. According to the study by Mansfield (1986), a

telecommunications, software and biotechnology, because in these industries there is an enormous gap between the cost of discovering or developing a new innovation and the ease with which innovations can be copied.[16] The authors of the study assert that small firms have been one of the drivers of innovation in these industries and these small firms partly rely on patents to attract either research partners or investment.[17] They assert also that European policy makers have reacted to the rise of a pro-patent era with both unease and resolve. The unease is based on the belief that the European innovation system has not been as successful as the American and Japanese systems. The resolve comes from the conviction that new IPR policies could enhance the competitiveness of European firms. European policy makers believe that appropriate changes to the European patent law could help to foster and maintain European competitiveness.[18]

The underlying idea of the European Commission's proposed directive on the patentability of computer software is the assumption that software patents motivate smaller developers to innovate.[19] An economic study[20] finds that "the patentability of computer program related inventions has helped the growth of computer program related industries in the States, in particular the growth of SMEs and independent software developers into sizeable indeed major companies". The study also finds "no evidence that European independent software developers have been unduly affected by

---

lack of patent protection would have prevented the development of 60% of pharmaceuticals, 38% of chemical inventions, 17% of machinery, 12 % of fabricated metals, 11% of electrical equipment, and had no effect at all on office equipment, motor vehicles, rubber and textiles.

[16] Anthony Arundel Merit, *Patents-the Viagra of innovation policy?*

[17] SMEs account for more than 99% of all European firms, 66% of all jobs and 65% of turnover in the European Community. See this study, 6.3 [4] Implications of the Utility Model Regime.

[18] European Commission, *Green Paper on Innovation*, and the *First Action Plan for Innovation in Europe*, 1998, cited by Adams and Tang, p. 92, n.7; Anthony Arundel Merit, *Patents-the Viagra of innovation policy?* European Commission, *Innovation Policy in a Knowledge-Based Economy.*

[19] *Promoting innovation through patents: Follow up to the Green Paper on the Community Patent and the Patent System in Europe*, Communication from the Commission to the Council, the European Parliament and the Economic and Social Committee, 1998.

[20] Robert Hart, *The Economic Impact of Patentability of Computer Programs*, at 3, 5.

the patent positions of large companies". According to this view, software patentability encourages innovation because firms may be able to protect their financial investments and attract venture capital.[21]

On the other hand, those who oppose extended protection argue that no incentive is necessary in order to encourage development, because (1) software developers are obliged to innovate constantly in order to survive and (2) the positive effects of networks already give a temporary monopoly to the inventors who are the first to put their ideas into practice following the principle of 'first mover takes all'.[22] Because of the need to innovate constantly there is no need for any further incentive. Unlike a tangible good like a car, software does not wear out and the functions at first are the same as those after several years, so that there is no chance of selling another copy of the software to the same person. In addition, they argue that if the exploitation of non-IPR appropriation (e.g. secrecy, lead-time advantages, technical complexity, complementary services, etc) already provides an adequate incentive to innovate, there is no reason to strengthen IPRs.[23] According to the article, *Do Stronger Patents Induce More Innovation? Evidence From The 1998 Japanese Patent Law Reforms,*[24] an expansion of patent scope induces generally modest innovative output and additional R&D effort. There is little evidence that this expansion of patent scope induced additional R&D effort by Japanese firms. The empirical evidence suggests that firm responsiveness to even significant changes in patent design is limited. Arundel[25]

---

[21] Adams and Tang, p. 2.
[22] Jean-Paul Smets-Solanes, *Stimulating competition and innovation in the information society*, www.pro-innovation.org 23 March 2001. See also *Competition Policy in the New High-Tech, Global Marketplace*, Volume I, A Report by Federal Trade Commission Staff, May 1996, Chapter 1. A greater benefit of being first is winning a place in consumers' hearts and minds.
[23] Anthony Arundel Merit, *Patents-the Viagra of innovation policy?*
[24] Mariko Sakakibara and Lee Branstetter, NBER Working Paper 7066, http://www.nber.org/papers/w7066 April 1999.
25 The author of *Patents-the Viagra of innovation policy?*

explains that the incentive provided by patents is relatively small on average, and less than the incentive provided by other appropriation methods. Those who oppose extended protection also assert that "bad patents",[26] such as BMPs, certainly increase costs for users and/or competitors.

However, when the life cycle of products is shortened and the speed with which competitors can turn out new products is enhanced, being first mover may no longer be a guarantee for success.[27] Furthermore it is a question whether inventors, especially those such as the pharmaceutical industry where innovation is slow and costly, would be willing to invest so much money and effort in their R&D of technology which, without a patent system, would not be able to be protected.

The TRIPs Agreement does not allow the exclusion of software in general from patentability.[28] However, there has been criticism of the American decision to allow the patenting of software.[29] Bessen and Maskin argue that strengthened patent protection ushered in a period of stagnant R&D among software and computer industries. They maintain that, in those industries, imitation can promote innovation and strong patents (long patents with broad scope) can hinder. This is because, they explain, in these industries innovation is both sequential and complementary.

On the other hand, there is an appreciation of the law on patentability of computer programs in the US, which has been regarded as having a positive influence on the development of the software industry in the US.[30]

---

26 Priceline's patent, "A reverse auction system", has been blamed by commentators as an example of bad patents.

27 US FTC, *Anticipating the 21st Century*, Chapter 1, pp. 25, 35.

28 Article 27(1) of the TRIPs Agreement requires that patents be available "in all fields of technologies, provided that they are new, involve an inventive step and are capable of industrial application".

29 James Bessen and Eric Maskin, *Sequential Innovation, Patents, and Imitation*, Working Paper Department of Economics, Massachusetts Institute of Technology, No. 00-01, January 2000 (afterwards, 'Bessen and Maskin, *Sequential Innovation*').

30 Communication of the European Commission to the European Council, the European Parliament and

By reviewing the advantages and disadvantages, this section of this study examines the desirability of patent and copyright protection of software and business methods to encourage innovation. To do this, it is necessary to discuss the characteristics of software, software industry, the Internet and e-commerce.

5.2 Characteristics of Software

The preliminary question for software patent protection must be what the characteristics of software are. Computer programs have a number of characteristics. First, the primary value in a program lies in behaviour, not in text.[31] Second, program text and behaviour are independent. Third, programs are, in fact, machines (entities that bring about useful results, i.e. behaviour). Fourth, the industrial designs embodied in programs are typically incremental.

[1] Text and Behaviour

The nature of software is intrinsically a hybrid combining both text and behaviour.[32] Software's text is its "literal expression", while software's behaviour has non-literal aspects. This literal expression of software is known as source code or object code, which are protected by copyright. Typical software customers never see the literal expression. The process of compiling converts source code into machine-readable object code. Actually almost all software is distributed in object code form, not in source code. Source code is like an architectural blueprint, which provides the

---

the Economic and Social Committee, 5 February 1999, COM (1999) 42.

[31] *A Manifesto*. Section 1.

underlying structural expression, while object code is more like the building itself.

Although the text of a program is designed to produce certain behaviour, program text and behaviour are generally independent.[33] Two programs with different texts can have completely equivalent behaviour. A second comer can develop a program having identical behaviour, but completely different text. The independence of text and behaviour is one important respect in which programs differ from other copyrighted works.[34]

The value of a program lies in behaviour rather than in text. As long as behaviour is the primary source of value in a program, programs with identical behaviour are market substitutes even if they have different text.

[2] Programs are Machines

Programs are machines whose medium of construction is text. Programs behave like machines. They produce useful behaviour like machines such as cars. Programs and physical machines are completely interchangeable: an electronic device that combined into a computer could deliver identical behaviour. Programs, like other machines, often work together with other programs or other machines to bring about specific results. Creating programs is a process of building and assembling functional elements, such as algorithms and data structures. Like physical machines, programs are often large and complex. Typical programs consist of text with hundreds of thousands of lines. The largest programs are roughly comparable in component count to some of the most complex mechanical devices. The behaviour of software, like the behaviour of other

---

[32] Paley, *A Model Software*.
[33] *A Manifesto*, Section 1.

machines, can be either utilitarian or fanciful. Program text is, like steel or plastic, a medium in which other works can be created. A novel device built in the medium of steel or plastic is patentable; an original sculpture built of steel or plastic is copyrightable. The medium in which the work is made does not determine the character of the creation. Likewise, the legal character of software should not be determined by the medium, program text. Programs can be regarded as virtual machines and a proper subject for protection.[35]

[3] Conceptual Metaphors

Programs generally rely on conceptual metaphors to organize behaviour. Programs often accomplish tasks by providing a conceptual metaphor. The metaphor brings about a synthetic reality, that is, "virtuality". For example, word-processing programs use the conceptual metaphor of paper to provide users with 'virtual paper'. The virtual paper can do work that ordinary paper cannot. The conceptual metaphor can change the user's experience of the task.

An innovative conceptual metaphor is one of the most valuable types of software innovation. For example, there are powerful conceptual metaphors in spreadsheet programs. Spreadsheet programs were the application programs that brought about the significant surge in the acquisition of PCs. The power of this conceptual metaphor created a revolution in the use of computers.

The effort to develop the software of such valuable tools should be protected by any legal regime. Although conceptual metaphors are important sources of innovation in

---

[34] It is difficult to imagine creating two pieces of music that have different notes but identical sound.
[35] Samuelson, *A Manifesto*, Section 1.

software products, because they are remote from the program text and abstract in character, they are unlikely to be protectable by copyright law.

[4] Programs are Industrial Compilations

Programs are industrial compilations. Program construction requires selection and arrangement of useful components so that the software machine produces its desired behaviour. Computer programs are compilations of sub-components. Larger programs are built from smaller programs.

Because programs are composite in character and programs behave, programs are compilations of behavioural components.[36] Each of these components is carefully designed and collected in order to produce a desired behaviour when they work together. Hence, programs are compilations of useful behaviour.

Since programs are machines, writing programs is an industrial design process similar to the design of physical machines. Designing involves a skilled effort to decompose the overall, complex task into a set of simpler sub-tasks, and to construct the interaction of these sub-tasks to produce useful behaviours.

[5] Incremental and Cumulative Innovation

Innovation in the case of programs is largely incremental and cumulative. Programmers commonly adopt software design elements and adapt them to a new context or set of

---

[36] One of the characteristics of modern software development is componentization. See this study, 2.5 Characteristics of Modern Software Development.

tasks.[37] In this way, programmers both contribute to and benefit from a cumulative innovation process. Most often innovation in program design does not rise to the level of invention, though occasionally it does.[38]

The products of software engineering almost always contain old and new elements. Some consist almost entirely of old elements. The innovation in such programs may lie where the known elements have been combined in a new and efficient manner, or where new elements have been combined with well-known elements.

5.3 The Nature of the Software Industry, the Internet and E-commerce

In order to understand the potential implications of software patents and BMPs, it is necessary first to understand the software industry, the nature of the Internet and e-commerce. After analysing key characteristics of the software industry and the Internet, the nature of e-commerce will be discussed from the point of view of buyers and sellers.

[1] The Nature of the Software Industry

The industry is currently characterized by rapid innovation, even if the innovation is primarily incremental.[39] The rapid innovation and strong competition verify that the software market is vibrant and successful.

---

[37] Software re-use involves integrating software components from existing software into the development of new software. See this study, 2.5. [4] Software Re-use.
[38] Randall M. Whitmeyer, *A Plea For Due Processes: Defining the Proper Scope of Patent Protection for Computer Software*, 85 Nw. U. L. Rev. 1103, 1131 (1991) (expressing the view that the overwhelming majority of software innovations would not meet the Patent Act's novelty and non-obviousness standards even if subject-matter hurdles could be overcome.)
[39] *A Manifesto*, Section 4.4.

The market is generally characterized by a large number of small companies.[40] This shows that innovation in this industry is often accomplished by small companies, and that any legal regime should be wary of its impact on small companies.

The market is maturing and barriers to entry are developing.[41] For example, the growing importance of a base of trained users indicates the maturation of the market, which in turn is becoming a barrier to new entry to major application product domains.

The established base of trained software users reinforces the need for interoperability. The need for interoperability means that there are natural incentives for many companies to share information. For example, new competitors should be able to handle data in the forms produced by their competitors in order to facilitate users' adoption of their new products.

[2] The Nature of the Internet[42]

(1) *Universality of access*

In the Internet, distance no longer plays a role. Regardless of location and time, any end user can gain access to the Internet, which can best be described as a 'network of networks'.[43] Consumption of the site by one consumer does not preclude or diminish consumption by another consumer.

(2) *High speed of information flow*

Large quantities of data can be transmitted, retrieved and processed within a matter of

---

[40] More than half (57%) had 15 or fewer employees. 71% had 30 or fewer employees. (See Massachusetts Computer Software Council, Inc., Software Industry 1993 Business Practices Survey 1019)
[41] *A Manifesto*, Section 4.4.
[42] Office of Fair Trading, *E-commerce and its Implications for Competition Policy*, Discussion Paper 1, a report compiled by the Frontier Economics Group for the Office of Fair Trading, OFT308, August 2000, (afterwards, 'OFT, E-commerce') p. 11; Christoph Engel, *The Internet and the Nation State*, presented at the Kiel meeting of the Society in March 1999.

seconds.

(3) *Interactivity*

Internet protocols facilitate considerable interactivity between websites and end-users. It is possible to link databases to the Internet, so that any end-user (with a browser and web connection) can access the information in the databases. Servers can display the information tailored to the individual customer's preferences.

(4) *Integration*

Just as it is possible to link servers at remote destinations, it is possible to link servers within a single organization. These 'Intranets' can be used as platforms for corporate management and information systems. Moreover, linking the servers of different organizations can create 'Extranets', which can be used for inter-business transactions.

(5) *Decentralization*

Since the Internet is not a single network but a network of networks, control of it is made more difficult and entry into the market of network providers is facilitated. To be an Internet provider needs a single server. The customers of this one provider are able to communicate with people and computers all over the world.

(6) *Timelessness*

The Internet lacks a culture of forgetfulness, since information on the Internet is available anytime. Search engines discover almost the same information even years later. Even if the originator erased the information from the original server, the information remains on the Internet due to mirror technology. This characteristic of the Internet makes it bridge long periods.

(7) *Cheapness*

To communicate via the Internet with people on the other side of the earth costs only

---

[43] OFT, E-commerce, pp. 11-12.

telephone charges. This is true for transmission costs to the next interface, for the activities of an Internet provider and for the services that can be used via the Internet.

(8) *Simplicity*

E-mail is fast and simple for an ordinary computer user. On the paper of conceptual metaphor, one can write an ordinary letter and simply send the letter by clicking the 'send' icon. Responses to incoming mail can be made within a few minutes using the 'reply' function. It is easy to exchange tens of letters a day with others via the Internet.

(9) *Digital transmission*

All Internet communication takes place digitally. Texts, sounds and images are transformed into 1/0 combinations. Since all these types of content can employ the same method of transmission, there is no more need for separate channels for data transmissions and all these transmission paths are in competition with one another. In addition, digital information can be attached to any chosen product. Images, sounds or video sequences can be added to home pages. Individual components can be compiled from diverse sources.

(10) *Wireless transmission*

Since information is communicated digitally, both wire and wireless transmission paths can be used. Wireless transmission opens up completely new possibilities. By carrying a small transmitter and receiver, one can control all household appliances remotely. Satellite transmission further increases the international impact of the Internet.

(11) *Secrecy*

Encryption serves as a tool for authentication purposes. It ensures that the transmitted contents really originate from the sender.

(12) *Anonymity*

Internet users can remain anonymous.

[3] The Nature of E-commerce

In the absence of a local customer base and physical sales, and in the presence of potentially low buyer switching costs, reputation, branding and customer loyalty may become increasingly important. The costs involved in developing these factors (reputation, branding and customer loyalty) may create significant first-mover advantages, and act as a barrier to later entrants. These costs relate to attracting customers to the site, building trust in the site, and establishing a brand name and customer base, which are crucial to the success of e-commerce.[44]

Online marketplaces are characterized by 'network effects'. The more participants a website has, the more useful it becomes to its participants. The value of a marketplace to its participants increases with the number of participants. For example, no buyer will wish to buy from a marketplace in which just one or two sellers are represented, if it can move to another marketplace where it can choose between many sellers. Likewise most sellers will wish to sell in the marketplace with the most buyers. In such a place, the strong players become stronger and the weak weaker. These network effects become further first-mover advantages and barriers to entry.

5.4 Desirability of Software Patenting

Traditionally, software has been protected by copyright. Small companies usually employ copyright protection because it is costless to acquire and maintain. As

information technology has developed, copyright protection for software has become inadequate because it is insufficient to protect the features or functions, i.e. the ideas embodied in a program. Patent protection for software is increasing.

A patent may claim a system for accomplishing a particular function, or a method for accomplishing the function. A patent may claim some features embodied in the program and processes accomplished by the program. In this regard the subject-matter of a patent is not the program as such, nor any manifestation of the program. On the contrary, the subject-matter of copyright is the actual instructions of the program. The subject-matter of copyright is the program itself or a manifestation of the program.

[1] Pros for Software Patents

The most significant feature of patent protection for software is that a patent can protect the idea or concept underlying the invention. The ideas or concepts embodied in computer programs may have great value. Furthermore, as the importance of business methods conducted on the Internet grows, protection for the business concept underlying the software, which has been developed for its specific purpose, becomes increasingly necessary. The growing necessity to protect software by patenting rather than copyright is extending the scope of patentability of software.

Software patenting protects against independent inventors, not just against copiers. This means that patents protect against independently developed programs that are based on the same concept. In addition, patent documents stimulate development, because the public can build on published software patents. Patents provide software developers with a potential source of income.

---

[44] OFT, E-commerce, Executive Summary, pp. 2-3, 43-46.

Advocates of extended software patentability assert that expanded protection provides stronger incentives for the generation and diffusion of new technologies.[45] Furthermore, the inadequacies in IP protection create loss of export sales and trade distortions in international trade.[46] Software patentability makes it worthwhile for investors to sink large resources into new or existing companies, and for new entrants to devote resources to R&D. A patent portfolio can also be used to bargain with companies for use of their patents.[47] A major advantage of a patent over copyright is that it can protect against competitors creating equivalent solutions.

[2] Cons for Software Patents

Patents have limited application in the protection of behaviour, because patents typically issue for particular methods of achieving results, rather than for results themselves. [48] A patent on a method of generating certain results cannot prevent the use of another method, even though those results are the principal source of value of the software. Hence, patents on methods would not protect behaviour, which is one of the primary entity of value in software. On the other hand, a patent with claims for any means of achieving particular results would inhibit competition in the development of useful program behaviours out of proportion to the innovation actually contributed by the applicant.[49]

---

[45] Michael B Wallerstein, Mary Ellen Mogee and Roberta A Schoen (eds.), *Global Dimension of Intellectual Property Rights in Science and Technology*, National Academy Press, Washington, DC., 1993.
[46] R Michael Gadbaw "*Intellectual Property and International Trade: Merger or Marriage of Convenience*", Vanderbilt Journal of Transnational Law, vol. 22, No. 2. 1989; Michael R Michael and T Richards (eds.) *Intellectual Property Rights: Global Consensus, Global Conflict?* Boulder, CO: Westview Press, 1988.
[47] Adams and Tang, p. 11.
[48] *A Manifesto,* Section 2.
[49] See, e.g., U.S. Patent No. 5,317,757 to Medicke & Posharow, issued 31 May 1994 (System and Method

Disadvantages can also be exhibited by the characteristics of software innovation. Cumulative, sequential innovation and re-use prevail in the software industry.[50] Software innovation typically proceeds via a mix of new coding, modifications to some existing modules or subroutines, and re-use of others.[51] Moreover, patterns of improvement and re-use are constrained to a substantial degree by the need to preserve interoperability between program, system and network components.[52] Interoperability constrains the range of options available to the second-comers. Patenting incremental innovations, which are not inventive, would not be appropriate for the economic goals of patent system, since a patent is given for a substantial contribution to the art.

In this respect, software resembles semiconductor chips whose industrial designs are rarely inventive. Chip products are vulnerable to rapid copying that undermines the innovators' opportunity to recoup investment, and that undermines the incentives to develop new chip designs. To provide proper incentives for semiconductor design, the US Congress passed the Semiconductor Chip Protection Act (SCPA) of 1984.[53]

According to the concept of 'sequential innovation' systems, which was generalized by Bessen and Maskin, software patents are not economically useful as far as the ideal form of organization for the software industry is not a monopoly. A regime without patents induces more efficient investment than one with patents.[54] They argue that when innovation is sequential and complementary, contrary to standard reasoning

---

for Finite State Machine Processing Using Action Vectors); U.S. Patent No. 5,105,184 to Pirani & Ekedal, issued 14 April 1992 (Methods for Displaying and Integrating Commercial Advertisements with Computer Software).

[50] See Cohen & Lemley, *Patent Scope*; Lemley, *Encouraging Software Reuse*.

[51] See this study, Chapter 2. The Development of Computer Software.

[52] See this study, 2.6 [3] Compatibility and Interoperability.

[53] 17 U.S.C. §§ 901-914 (1988). However, the US Congress has failed to keep pace with rapidly changing technology and thereby has left the statute essentially irrelevant to present semiconductor technology. Some of the basic definitions of the Act are already obsolete and important parts of mask work technology are not protected by the legislation. Kenneth W. Dam, *Intellectual Property in an Age of Software and Biotechnology*, Chicago Working Paper in Law & Economics, May 1995.

[54] Bessen and Maskin, *Sequential Innovation*.

about patents and imitation, imitation becomes a spur to innovation and strong patents become an impediment. Since competition of many independent developers is preferable in order to stimulate innovation, the 'sequential innovation' model gives a theoretical demonstration of the harmful effects of software patents on innovation. Bessen and Maskin correlated development of innovation in the US with the growth of the number of patent applications, and showed that the increase in software patents and the shift towards more patentability led to a fall in expenditure on R&D and less productivity. They concluded that patents preserve innovation incentives in a static world; however, in a dynamic world, firms may have plenty of incentive to innovate without patents and patents may constrict complementary innovation. In this respect, they suggested that copyright may have achieved a better balance than patent protection, and that a patent system with limited patent breadth may offer a better balance.

Based on the characteristics of sequential innovations, opponents of software patenting argue that almost all authors of software will involuntarily infringe a software patent when they publish their software, because typical software comprises several thousand different processes (and major software, several tens of thousands) and depends on earlier programmers' solution to a problem. It is impossible to check to ensure that none of these processes infringes one of the 100,000 software patents already granted.

Since software development requires relatively less investment of time and money and the barrier to innovation is so small in the software sector, the absence of patents does not discourage innovation. In addition, the economic life of a software innovation is normally quite short. It is much shorter than the 20-year term conferred by the patent law. A software patent expanded to cover later improvements will exert control over

many more generations of improvements than a conventional patent with a longer effective term. This means that the market-distorting effect of a software patent will be substantially greater than that of other types of patents.[55] Software patents would also reserve for first-comers the rights to rule the market. To a greater degree than in other areas, second-comers would need permission to develop and market their innovations.

To be patented an invention must fulfil certain criteria: novelty, inventiveness and capability of industrial application. Computer programs are generally complex. Furthermore, application documents for software patents are highly complicated. Due to these requirements and complexities, the cost of using the patent system, i.e. filing, maintaining and defending a patent, is high, particularly for SMEs.[56] Litigation also imposes much greater difficulty on SMEs. On the other hand, large companies can own many patents with their abundant resources and use them to cross-license. If a small company tries to use a patent to protect itself against a large company, the large company can find patents among its collection which the small company is infringing, and the large company can require a cross-licence. While the large companies thus indirectly benefit from the patent system, SMEs are threatened by the system. This may lead to the lack of innovation and competition, since small businesses are the significant source of innovation.[57]

Judging from the general conclusions of the research produced by the *Intellectual*

---

[55] Cohen & Lemley, *Patent Scope*.

[56] Special consideration should be given to SMEs. In Japan, SMEs contribute 99% of all establishments, 52% of output, and 72% of employment. In Korea, they are responsible for 90% of all establishments, 33% of output and 51% of employment. See *WIPO Milan Forum on Intellectual Property and Small and Medium-sized Enterprises*, Milan, Italy, 9-10 February 2001, WIPO/IP/MIL/01/4 (A). Even though SMEs, in the US, receive less than 4% of Federal support for research, they produce more than half of the innovations and get close to 40% of all patents (State of Small Business Report, 1997). See William Kingston, *Meeting Nelson's Concerns about Intellectual Property*, available at http://www.business.auc.dk/druid/conferences/nw/paper1/kingston.pdf accessed 13 August 2001.

[57] US FTC, *Anticipating the 21st Century*, Chapter 5, pp. 1, 2. See also this study, 6.3 [4] Implications of the Utility Model Regime.

Property Initiative, there is little empirical evidence to support the view that SMEs would derive any additional benefit from expanded patent and copyright systems. The significant conclusions derived from the research are the following:[58]

- SMEs generally rely on copyright for their software creations.
- SMEs patent less, since they find the system complicated and expensive, and they do not think of patents as conferring any particular advantage for their software products. The indifference of SMEs to the patent system is much worse than we expected.
- Very few of the firms attached any particular value to the patent system as a means of protection, although these firms regarded themselves as innovative.
- Importantly, the majority of SMEs (87%) claimed that they would have developed their inventions (which were patented) without a patent.
- Because of the lack of resources, it is difficult for SMEs to defend patents.
- SMEs do not particularly use patent information for their innovations.
- SMEs employ, in addition to copyright, informal methods of protection (e.g. encryption and passwords, trust relationships, market niche, first mover advantage, and secrecy) as effective methods of protection.
- SMEs feel that amendments to patent law will make it more difficult for them to cope with developments, because of their inability to keep up with them.
- SMEs tend to be more concerned with developing their products and getting them to market in the shortest possible time, than protecting them formally.

---

[58] See Adams and Tang, pp. vi-vii, 15-16; Robert A. Blackburn, *Intellectual Property and the Small and Medium Enterprise*, http://info.sm.umist.ac.uk/esrcip/Projects/15253004.htm accessed 8 October 2001; Stuart Macdonald, *Protection or Dissemination? The Contribution of the Patent System to Innovation of SMEs* http://info.sm.umist.ac.uk/esrcip/Projects/15253021.htm accessed 8 October 2001. The research was funded by the UK Economic and Social Research Council, the UK Department of Trade and Industry (DTI) and the Intellectual Property Institute (London). The program was conducted between 1996-1999.

These concerns are from the two pressures that are the rapid developments in the software industry and the speedy obsolescence of software products.

The introduction of software patenting may also increase secrecy of practical technical knowledge and hinder the sharing of knowledge. It is because the publication of the source code facilitates a competitor's search for patent infringements (whereas the publication of the binary code prevents a competitor's search for patent infringements due to the prohibition on decompiling),[59] that publishers keep source codes secret to reduce patent infringement lawsuits.[60] In this respect, open source software has several advantages.[61] Among them, accessibility to the source code and the right to modify it are the most significant ones. Another involves the right to use the software in any way. The absence of the exercise of proprietary rights allows for various uses and improvements of the products. There is no fear of being held hostage by a proprietary software company. Disadvantages of open source software are the lack of guarantee that development will happen, and the possibility of being accessed by patent holders who are trying to detect infringement through the accessible source code.

The subject-matter of software-related inventions is changing as technology develops. Due to this, software patents whose scopes are uncertain are being granted. Uncertainty is bad for business, and makes it difficult to decide the best strategy to pursue. Subject-matter uncertainty also makes it expensive to search, to analyse and to fight in court.

Software patenting has inappropriate aspects in that relevant prior art is not

---

[59] Decompiling software means analysing its working principles through reverse engineering. The prohibition on decompiling applies in Europe and the US. In Japan, this prohibition only applies in practice to American software, following bilateral agreements. For information on decompiling, see www.softpanorama.org/SE/reverse_engineering_link1.shtml.

[60] Jean-Paul Smets-Solanes.

[61] Adams and Tang, pp. 12-13.

disclosed sufficiently, because patent protection for software has been limited, and software developers have kept secret the software source codes that they have developed. Also, because the vast majority of software innovation takes place outside traditional research institutions, many software improvements are not recorded in the formal system of technical documentation. Software innovations exist in the source code of commercial products and services that are available to customers.[62] This source code is difficult to catalog or search for ideas. This trend results in insufficient published prior art, which makes it difficult to search for prior art and to examine patent applications properly. In addition, since patent examiners work under severe time constraints, particularly in the software-related art units, they do not have enough time to spend searching for appropriate software prior art that is scattered throughout the patent classification (e.g. International Patent Classification).[63] Software patents had the tendency to be classified according to the field in which the software will eventually be used (e.g. game machine, ovens, washing machine), rather than according to the nature of the software invention. This in turn makes it much harder for examiners to find relevant prior art. As a result, software patents are more likely to receive a broader scope than they deserve.

5.5 Copyright Protection for Software[64]

According to the US statutory definition of 'computer program', program texts are clearly protected.[65] Copyright does not protect the results (i.e. behaviour) brought about

---

[62] Cohen & Lemley, *Patent Scope*.
[63] Ibid.
[64] H.W.A.M. Hanneman pp. 4-5.
[65] See 17 U.S.C. § 101 (1988) (defining 'computer program' [as] a set of statements or instructions to be used directly or indirectly in a computer in order to bring about a certain result).

by the execution of program.[66] Only when program behaviour is expressive in a traditional copyright sense, does copyright protect program behaviour. When the execution of program instruction results in a series of pictures combined with text and sounds, copyright protection for the audiovisual work is appropriate.[67]

Nearly all programs are copyrighted. Traditionally, it has been thought that copyrighting applies to the expression of ideas. Copyright protects only the specific form in which the idea is expressed. It protects the form of expression, i.e. source code and object code, from duplication or close imitation. Copyright may be applied to the program's structure, sequence and organization, and some elements of the user-interface.[68] Copyright prohibits the users of a software program from making copies of it without permission.[69] It prevents one company from appropriating another company's work and selling it as its own. However, it does not prevent other programmers from using algorithms or techniques contained in the program. A single software technique can be implemented in different ways to do totally different jobs.

Copyright law forbids protection of "any idea, procedure, process, system, method of operation, concept, principle, or discovery, regardless of the form in which it is described..."[70] Copyright cannot protect underlying functions, methods, ideas, systems, or algorithms. Software's *utility* is protected under patent law. In limited situations, patents have been granted on aspects of computer software comprising those forbidden

---

[66] *A Manifesto*, Section 2, p. 8.
[67] See, e.g. *Stern Elec., Inc. v. Kaufman*, 669 F.2d 852 (2d Cir. 1982) (audiovisual copyright in videogame's repetitive sequence).
[68] www.library.carleton.edu/staff/michael/acarticle.html accessed 7 January 2002.
[69] The US recognizes the following seven exclusive rights to a copyright holder: 1) reproductive right, 2) adaptive right (to produce derivative works based on the copyrighted work), 3) distribution right, 4) performance right, 5) display right, 6) attribution right (to claim authorship of the work and to prevent the use of his/her name as the author of a work he/she did not create), and 7) integrity right (to prevent the use of his/her name as the author of a distorted version of the work, to prevent intentional distortion of the work, and to prevent destruction of the work).
[70] Copyright Act of 1976 § 102(b), 17 U.S.C. § 102(b) (1993).

by copyright law above.

Copyright in a computer program is not infringed by the acts done for the purpose of studying the ideas or principles underlying a program, the reproduction or translation of code in order to achieve the interoperability of an independently-developed computer program.[71]

In order to pursue the best policy to encourage innovation, it is also necessary to examine advantages and disadvantages of copyright protection for software.

[1] Advantages of Copyright Protection

Copyright attaches to representations (expressions) of a scientific or technical nature. It is acquired automatically and therefore requires no expenditure of time, effort or money. It lasts at least 50 years p.m.a.[72] A large number of firms use copyright as their main protection, because it is cheap, automatic and effective. It is convenient for small companies.

Software copyright efficiently protects source code secrecy through the prohibition on decompiling which is guaranteed by law in Europe or by copyright licensing contracts in the US. In a system without software patents, publishers who do not want their technical knowledge to benefit competitors can keep their software source code secret. Source code secrecy allows publishers to conceal possible copyright infringements.[73]

---

[71] *Directive on the Patentability 2002/0047 (COD)*, Article 6, pp. 8, 21; Directive 91/250/EEC Articles 5 and 6.
[72] Berne Convention (1971 text) Art 7.
[73] Jean-Paul Smets-Solanes, *Stimulating competition and innovation.*

[2] Disadvantages of Copyright Protection

The dual nature of programs has created conceptual difficulties for copyright law.[74] As explained above, text and behaviour are largely independent, so that protecting program texts cannot prevent competitors from copying valuable program behaviour. The ability to copy valuable behaviour legally would sharply reduce incentives for innovation.

Once it is recognized that computer programs are machines whose medium of construction is text, it becomes obvious that copyright protection for program behaviour is inappropriate. As copyright law does not protect the behaviour of physical machines (nor their internal design), it does not protect program behaviour.

For this reason, it is argued that copyright protection is not entirely appropriate to protect software, because it offers relatively narrow protection. The most important characteristics embodied in software, if expressed in another way, cannot be subject to infringement proceedings. Copyright protects only the specific form in which the idea or the concept is expressed. Everybody can use the concept or the idea itself. In addition, copyright does not protect against independently developed software. If anyone else independently develops the same program he may use it freely.

Opponents also argue that the copyright protection period is excessive considering the short life of software product and 20-year maximum period of patent protection.[75]

5.6 Desirability of Business Method Patenting

In spite of many debates on the patentability of business methods, the number of

---

[74] *A Manifesto*, Section 2.

[75] They argue that copyright gives inappropriate terms of protection which can be up to 120 years. See

applications with claims for software-based "business processes or steps for doing something" is increasing in the US.[76] The USPTO has granted thousands of BMPs, and American companies are filing applications for their business methods to the EPO. According to a recent study done by Olswang and Oxford University, more than 400 BMP applications have been filed at the EPO during the period 1996-1999. US companies accounted for 52% of these applications.[77] This trend of filing for BMPs is likely to continue unabated in the light of their perceived importance for the development of e-commerce. According to the material on the website of Walker Digital, Inc,.:

> *Walker Digital holds 50 US patents, and has approximately 300 patents pending. Walker Digital's intellectual property portfolio reflects the company's commitment to developing highly innovative new technology and Internet-based solutions to business problems.*[78]

As mentioned above, advocates of extended software patentability assert that expanded protection provides stronger incentives for the generation and diffusion of new technologies. However, BMPs have given rise to complaints that they will stifle e-commerce.[79] The traditional arguments against patenting business methods are that innovations in ways of doing business are not like developing new drugs or pesticides which require much R&D investment. There is also widespread concern that several of these patents are trivial, non-inventive and obvious. These complaints have highlighted the inadequacy of the current examination. Gregory Aharonian laments:[80]

> *There are many fears of future patent litigation... there is another danger, more psychological than financial... due to the trivial and obnoxious*

William Kingston, *Meeting Nelson's Concerns about Intellectual Property.*

[76] Adams and Tang, p. 9, 10.

[77] 20% from the UK, Germany and France. Adams and Tang, p. 7.

[78] www.walkerdigital.com accessed 18 August 2000 cited by Adams and Tang, p. 11.

[79] Adams and Tang, p. 10.

[80] "Random bits: Gregory Ahronian: Walker Asset trivial/obnoxious business method patents," http://lists.essential.org/pipermail/random-bits/2000-April/000143.htm accessed 13 August 2000, cited by Adams and Tang, p. 94, n.46.

obviousness of many of these patents.

Those opposing the extension of patentability of software argue for limiting any negative effects from "bad patents" lacking in novelty or non-obviousness.[81]

According to the UKPO's Conclusions,[82] there is no sign of a want of innovation in computer-implemented business methods, nor was there in the US before business methods became patentable in 1998. The UK Government concluded that those who favour some form of patentability for business methods have not provided the evidence that it would increase innovation. It also concluded that unless and until the evidence is available, ways of doing business should remain unpatentable.[83]

As Lawrence Lessig advises,[84] it is necessary for policy makers to strike a balance and question whether BMPs will induce more innovation. The solution of this issue may be available through discussing the advantages and disadvantages of patenting business methods.

[1] Advantages and Necessity of Protecting Business Methods[85]

In the past, the courts have declined to sustain patents on a new business method,[86] on the ground that business methods were mere concepts without any connection to physical objects. These days, however, every new concept in financial services is carried out with computers because business concepts can be implemented by

---

[81] Adams and Tang, p. 11.
[82] www.patent.gov.uk/about/consultations/conclusions.htm accessed 11 May 2001, UKPO, Conclusions to the Consultation on "*Should Patents be Granted for Computer Software or Ways of Doing Business?*"
[83] However, it is not clear that whether "ways of doing business" in the questionnaire include computer-implemented business methods.
[84] Lawrence Lessig, "*Europe's me-too patent law*", *FT.com,* 11 July 2000, cited by Adams and Tang, p. 95, n.50.
[85] Robert P. Merges, *Patent Law*, n.p., 1997, pp. 152-155.
[86] *In re Sterling,* 70 F.2d 910 (CCPA 1934), 'a new bookkeeping system employing novel checkbooks', the court refused patent protection because the physical structure presented no novelty; *In re Wait,* 73 F.2d

computers. Without computers, the complicated, voluminous transactions in stock markets or banks would be impossible. The innovations in these fields must be performed using computers.

The necessity of patenting business methods can also be supported by three economic rationales. First, the economic environment has changed and the cost of creating an innovative financial service or instrument is rising and old rules do not provide enough incentive to innovate in this field. Second, it is unfair to exclude these innovations from patent protection on the basis of subject-matter, because people in these fields work as hard as in any other field where patents are permitted. Third, the financial services sector has performed admirably, but it would have done, and will do, even better in the new world of patents. Investments will be made without fear of piracy, because companies now know that the innovations resulting from their investments will be protected.

[2] Undesirable Aspects of Business Method Patenting[87]

There are two difficulties in patent protection for business methods. One relates to the quality of the patents, the other to the economy of competition.

The USPTO has been criticized because it has issued patents on mundane business inventions.[88] This is mainly because prior art and examination standards for BMIs are not properly established. Even information about well-known methods may not be at the examiners' hand, because business methods exist mainly in practice. In addition, in the

---

982 (CCPA 1934). 'A system for conducting commodities trades at a distance without using brokers'.

[87] Rochelle Cooper Dreyfuss, *Are Business Method Patents Bad for Business?* Santa Clara Computer and High Technology Law Journal, Vol. 16(2), March 2000. This paper is also available at: http://papers.ssrn.com/paper.taf?abstract_id=219574.

[88] US Patent 5,794,207.

case of BMIs it is difficult to determine what is truly novel and non-obvious. Thus, business methods cannot be examined effectively and this results in 'bad patents'.

It is argued that taking a well-known method of doing business and, for the first time, carrying out the business method online cannot be an invention. It is also argued that it does not make sense to grant a patent to one programmer on something that is obvious to many others. Nor would such a patent be valid.

In the light of competition, opponents of BMPs argue that big companies will have almost all of the business-related patents and such patents will stifle competition by the elimination of small companies. They argue that business methods must be in the public domain, because they are fundamental to the economy like the laws of nature. They argue that many business methods had been developed even though these inventions were unpatentable, and that because such patents will benefit only patentees against other software developers, such patents will impinge adversely on innovation: Furthermore, because patents can be granted to well-known business methods transferred from the real world to the Internet, and because modern businesses are generally conducted with computers and through the Internet, the claims of the patents are so broad that they may cover even the original business methods themselves. These criticisms are reasonable in that, while a patent is given to an inventor in return for his investment, broader protection than for his contribution results in a cost to the society the inventor belongs to. Therefore, it must be examined whether the transformation from the real world to the Internet is so significant as to be patentable, even though new applications of hardware are found in the transferring.

Patenting business methods not only distorts the competitive marketplace, but also cripples the ability of Adam Smith's "unseen hand" to allocate resources to their highest

and best uses.[89] The problem of invalid BMPs cannot be overcome by court's post-invalidation because the effects of these patents endure beyond invalidation. Patents covering e-commerce create network effects. Once a customer is locked into a system, invalidation of the patent no longer matters because the customer will not switch.[90] For example, Amazon analyses the information it has accumulated. It compares each customer's purchases with other information in its database to predict other books the buyer will enjoy. The accuracy of these predictions depends directly on the size of the database. Thus, the bigger the network of Amazon, the more valuable the network is. If the patent is invalidated, rival sites will be able to offer 'one-click'. But their networks are not so big yet that customers are unwilling to use them. They do not want to shop at a site that needs additional work and provides less value. Amazon may, in short, monopolize the Internet bookstore not because it is the best bookstore, but because it had the patent on 'one-click'. This adverse effect is much more severe when the patent is valid, or when it remains valid for very long period of time.

Lawrence Lessig worries that while increasing patentability may increase incentives, it certainly increases costs.[91] Lessig advises that before Europe expands the patentability of software, it is necessary to study whether there is any good economic reason to believe that software patents will induce more innovation. Lessig argues that if patenting software will induce more innovation in software development, then proponents should demonstrate it through convincing evidence. He advises that Europe should wait until they do.

Eurolinux fears that an expanded patent system would create increased costs for

---

[89] http://commdocs.house.gov/committees/judiciary/hju63845.000/hju63845_0.HTM accessed 14 January 2001.

[90] As for network effects, see this study, 2.6 [2] Standardization and Innovation Trade-offs.

[91] Lawrence Lessig, "*Europe's me-too patent law*", *FT.com*, 11 July 2000, cited by Adams and Tang, p. 95, n.50.

users and potentially restrict innovation. Jacques Le Marois[92] asserts that software publishers and innovative Internet business in the US constantly face the risk of a patent war, just because obvious techniques were granted patents.[93] Likewise, Jean-Francois Abramatic, president of the World Wide Web Consortium also has expressed concern with the evolving patent system.[94]

5.7 Conclusion

A vast amount of innovative software development has been taken place without patent protection. Much of it is being developed by individuals and SMEs. Extending patentability would impose a major burden on them because they would have to divert time and effort into making sure they are not infringing patents, and seeking and enforcing them.[95] It is necessary to consider whether extending the scope of software patents will work properly to produce innovative technology, because overly broad protection will stifle competition and result in a cost to the public,[96] while narrow protection will discourage inventors for innovating. Therefore, it should be assessed whether innovations are given protections in proportion to the contribution to the society the invention will make.[97]

In conclusion, software protection by existing legal regimes impedes follow-on

---

[92] President of MandrakeSoft, a software company based in France.
[93] www.petition.eurolinux.org/reference accessed 13 August 2000, cited by Adams and Tang, p. 95, n.54.
[94] www.Eurolinux.org accessed 13 August 2000, cited by Adams and Tang, p. 95, n.56.
[95] www.patent.gov.uk/about/consultations/conclusions.htm.
[96] If follow-on innovators are hindered by overly broad patents, there will be an inhibition on innovation and a risk of monopolistic profits being made at the expense of innovation.
[97] It is important to limit the patent system to those fields where the benefits will outweigh the disadvantages. This is the reason why patents have been confined to technological inventions. Ways of doing business and computer software as such that do not give rise to a 'technical effect' have not been patentable. See The UK Government's Conclusions to the result of the 1 November 2000 public consultation initiated by the UKPO. Available at www.patent.gov.uk/about/consultations/conclusions.htm.

innovations because they are based on the property right rules.[98] Under the liability regime, follow-on innovators can use the first comer's innovation only if they are willing to pay a certain royalty to the first comer.[99] Basic concepts of a new legal system would be found among liability regimes rather than exclusive proprietary ones. The new regime should solve the critical issue of the relationship between the first comer and second comers in sequential innovation, i.e. encouraging innovation without impeding follow-on innovations.

---

[98] A property entitlement or right precludes third parties from appropriating the object of protection, whereas a liability rule regulates the means by which they can engage in certain potentially harmful acts on certain conditions. For example, if one has rightful possession of something such as a car or a house under an exclusive property right, another person ordinarily cannot take it without permission, but under a liability rule, others may engage in acts that create risks of harm and thus constitute probabilistic invasions of property interests, while obligating them to pay damages for harm under specified circumstances. Louis Kaplow and Steven Shavell, *Property Rules Versus Liability Rules: An Economic Analysis*, 109 Harv. L. Rev. 713, 716 (1996).

[99] This lowers transactions costs and reduces undesirable social behaviour such as free riding appropriation.

CHAPTER 6

Alternative Proposals

6.1 A Market-Oriented Legal Regime

[1] Introduction

Considering the fact that the primary purpose of the IPR system is to encourage technological innovation and the transfer and dissemination of technology,[1] a new legal regime to provide appropriate protection for software should be a market-based (or market-oriented) one, in which innovation and dissemination of technology occur naturally. Under the regime, innovations should build on past innovations and incentives for innovations today should not stifle future innovations.

The authors[2] of '*A Manifesto Concerning the Legal Protection of Computer Programs*', have concluded that while copyright law can provide appropriate protection for some aspects of computer programs, other valuable aspects of programs (e.g. the useful behaviour generated when programs are in operation and the industrial design to produce this behaviour) are so vulnerable to rapid imitation that, if left unchecked, it would undermine incentives to invest in software development.[3] The authors of *A Manifesto* oppose efforts to expand the boundaries of existing legal regimes to protect these aspects of programs. They suggest that a *sui generis* approach to legal protection of computer programs is required. They explain why it is desirable to take a market-

---

[1] Article 7 of the TRIPs.

[2] Authors of '*A Manifesto Concerning the Legal Protection of Computer Programs*', 94 Colum. L. Rev. 2308 (1994). Pamela Samuelson, Randall Davis, Mitchell D. Kapor, and J.H. Reichman (afterwards, the authors of *A Manifesto*).

[3] *A Manifesto*, Introduction.

oriented approach to providing legal protection to these aspects of software. A market-oriented legal regime, they explain, needs criteria to assess when market failure is likely to occur.[4] They suggest three principal factors to judge the possibility of market failure from rapid copying: (1) the nature and size of the software entity (or component) that has been imitated; (2) the second comers' access means to the innovation and the degree of dependence of the second comers' product; and (3) the degree of similarity between the first and second products.[5]

Market failure is likely if (1) the nature and size of the entity imitated is substantial, (2) the second comers' development is rapid, easy and highly dependent on the first comer's product, and (3) the degree of similarity approaches identicality, and the second comers' market is proximate to that of the first comer.[6]

To provide suitable protection for computer programs, which do not fit neatly within the traditional forms of IP, the authors of *A Manifesto* discuss dimensions of the market-oriented legal regime, especially focussing on the entity dimension of software.[7]

[2] Primary Dimensions of the Market-Oriented Legal Regime

[a] Entity Dimension

The degree of legal protection to avoid market failure corresponds to the extensiveness of the taking of the program. According to the market-based legal regime, software entities are classified as follows:

---

[4] Ibid., Section 5.
[5] Ibid., Introduction.
[6] Ibid., Section 5.
[7] It is important to determine what is or is not protected.

a. Program code

b. Program compilation (program behaviour or industrial design for producing behaviour)

c. Subcompilations (a portion of a program's behaviour)

d. Algorithms[8]

e. Features

Program code

Program code, whether in source or object form, is unquestionably a valuable aspect of a program. Program code which embodies all of the valuable behaviour of the program, is now protected by copyright law in almost all jurisdictions.

The problem with exact copying of object code is that the copyist acquires behavioural equivalence at no cost and with no effort, resulting in the most serious danger of market failure because it undermines opportunities for the developer to recoup its R&D costs. Unless the situation is corrected, investments in innovation cannot be justified.

Program compilations and subcompilations

Copying a program compilation (e.g. program behaviour or the industrial design for bringing about behaviour) is also problematic because a second comer can generate a market substitute with relatively trivial effort. This kind of appropriation is less likely to have market-destructive effects than code copying since it requires *some* effort.

Copying a program's subcompilations (e.g. a portion of a program's behaviour) is generally less problematic than copying the program compilation because a second

---

[8] An "algorithm" was defined as "a procedure for solving a given type of mathematical problem". *Gottschalk v. Benson*, 409 U.S. 63, 65 (1972).

comer who copies only a portion of a program's behaviour may not offer a market substitute without expending considerable effort to design other aspects of its program. Nonetheless, some legal protection for subcompilations is required to provide innovators with enough opportunity to recoup their investment in innovation.

The extent of protection of these entities depends largely on the following:

- how extensive the appropriation is.
- how easy (or difficult) the appropriation was.
- how similar between the compilations embodied in the two products are.
- how quickly a new product embodying the appropriated innovation can enter the market.
- how proximate the markets of the two products are.
- how much the two products cost.

Algorithm and features

Algorithm and features may also be in need of some protection, although a second comer's copying them is less likely to have a destructive impact on the software market than copying code or program compilation has.

Since algorithms determine the behaviour of the programs that embody them, they can be protected within an industrial compilation framework. As long as algorithm innovation is valuable as well as vulnerable to quick and easy appropriation, a limited artificial lead-time may be necessary to stimulate investments for R&D.

Features, like algorithms, tend to be regarded as individual components of software products, although they are often composites of elements. Features that consist of only one or a small number of elements are probably too small compared to the

software product as a whole to affect investment incentives. Therefore, these features might be exempt from regulation in a market-oriented legal regime, though complex features might be regarded as subcompilations and be protected against market-destructive appropriations.

The firm that introduces a valuable feature to the market will have some natural lead-time protection due to its being first, because it may take a year or so for competitors to bring a competing product with the same feature into the market. While competitors are busy copying the feature to introduce it into later versions of their own programs, the originator may be adding other new features to its products. Thus, the new features give the developer a new lead-time advantage over those who have developed to catch up with the previous innovation.

However, considering the fact that recent software copyright lawsuits[9] have focused on the copying of features as a basis for infringement claims, a market-oriented protection to features is required to give the originators some artificial lead-time.

[b] Access Dimension

The authors of *A Manifesto* distinguish types of dependent creation by the means through which a second comer gains access to the know-how in the originator's program. Means of access can affect the difficulty of imitating an innovation, and therefore, the speed, cost, and market effect of dependent creation. They classify means of access as follows:[10]

---

[9] *Lotus Dev. Corp. v. Borland Int'l, Inc.*, 799 F. Supp. 203, 205 (D. Mass.1992) (complaining about copying of macro feature); *John Richardson Computers, Ltd. v. Flanders* [1993] F.S.R. 497, 515 (Ch.) (British case complaining of copying of certain features of program for producing pharmaceutical prescriptions).

[10] *A Manifesto*, Section 5.

a. Dependent creation by means of:

(a) exact duplication of code

(b) abuse of access to nonpublic information

(c) decompilation

(d) detailed study of program externals

(e) other observations about the program or its functionality

b. Substantially similar program compilations

c. Independent creation (no access)

In a market-based legal regime, the lawfulness of different forms of reverse engineering and dependent creation will be assessed by the extent of the effort of second comers to acquire behavioural equivalence compared with the costs of initial development. Unless a given form would induce market failure, it would be lawful.

As dependent creation and reverse engineering are well-accepted practices in all technological fields, dependent creation of software is desirable unless it has market-destructive consequences. However, the standard rules about reverse engineering and imitative copying of unpatented technical innovation need to be adapted in relation to computer software since the software innovations are much more vulnerable to trivial acquisition of equivalence. To judge the legitimacy of various means of reverse engineering, it is also necessary to evaluate the market impact of various forms of dependent creation of software.

Duplication of code

If program code embodied in products could be freely appropriated, it would result in market-destructive appropriations of program innovations and underinvestment in the

development of programs.

Abuse of access

If an imitator abuses access to source code, flow charts, or other documentation in breach of a confidential relationship or an enforceable contractual restriction, this leads to market-destructive dependent creation of software. Trade secrecy and contract law protect against this form of dependent creation. The trade secret holder can get an injunction to restore the lead-time advantage, which the holder would have had if abuse of access had not occurred.

Decompilation

Under the principles of trade secrecy and patent law, purchasers have been free to disassemble a marketed product embodying an unpatented innovation. If the disassembler thereafter devised another product embodying the innovation and sold it in competition with the original, the innovator would have no legal recourse.

However, in the case of software, decompilation should be regulated by the law, because, through decompilation, an imitator can create a virtually identical program which would cost significantly less than developing the decompiled program, and he can acquire behavioural equivalence with trivial effort.[11]

Nonetheless, decompilation hardly presents a serious threat of market-destructive effects, because it is very difficult and time-consuming.[12] In addition, even after the decompiler recognizes something useful from the decompilation, he must still make an effort to embody it in a differently expressed program. Thus, decompilation tends to be

---

[11] Ibid., Section 5.3.4.

[12] The larger and more complex the program, the more difficult it is to decompile it.

undertaken when it is the only way (e.g. when the licence fee is prohibitively expensive) to get access to essential information. As a consequence, software developers can maintain as trade secrets much of the know-how embodied in their programs. However, as reverse-engineering technology improves, the know-how embodied in program internals may eventually become vulnerable to rapid appropriation.

The authors of *A Manifesto* maintain that a complete ban of decompilation would be contrary to competition law, and limit re-use of information about incremental innovations. It would also result in wasteful replications of effort in the society.

Detailed study of program externals

Studying a program in operation can disclose information about its internal construction. For example, through watching a spreadsheet program in operation, a skilled programmer may discover the algorithm that must have been used to do its recalculation. It is known as "black box testing". Although virtually no one seems to argue that the black box testing ought to be prohibited as illegal form of reverse engineering, the authors of *A Manifesto* think that it can have market-destructive effects because of the relative ease and low cost with which it can be carried out. A functionally identical program in the market will undercut the opportunity of the first developer to recoup his investment.

Independent creation

Software developers often develop products independently for reasons other than the difficulty of reverse engineering. They may prefer their own solutions to a technological problem.

[c] Similarity Dimension

The degree of similarity between the first and second comers' products and the markets also influences in judging the potential for market-destructive effects. The authors of *A Manifesto* classified the degree of similarities in accordance with their market impacts:

a. Exact duplication of code

b. Clones (and near-clones) of (internal or external) compilations

c. Partial clones (Clones of subcompilations)

d. Substantially similar program compilations

   (a) without improvements

   (b) with improvements

e. Substantially different program compilations

   (a) migration of program elements to different markets

   (b) interoperating programs

   (c) add-on programs

f. Programs having the same general functionality but different particularized functionality

In general, the less extensive the similarity and the less proximate the markets of the producers, the lower is the potential for market-destructive effects, and therefore, the less need for legal regulation exists. When the second comer's program has the same general functionality but different particularized functionality, there is no potential for market failure.[13]

---

[13] *A Manifesto*, Section 5.

While the copyright case law tends to recognize only two kinds of similarities among programs: (1) literal similarities involving the copying of code, and (2) "nonliteral" similarities (as explained below), the authors distinguish a broader range of similarities among programs and program elements, considering that each has a different potential to cause market-destructive effects.

Exact duplication of code

Exact duplications of code have a high potential for market-destructive effects, because these are the most trivial way to acquire functional equivalence with another program. Whether the identity in code is complete or only partial, code copying is an easy case which copyright forbids. The result obtained under copyright law is consistent with the market-preserving principles.

However, in the case of programs which are similar only at high levels of abstraction, such as in their general purpose or function, if a second comer implements these functions in a very different way, the similarities present no danger of market failure. Both competition and innovation will be enhanced by the appearance of different products that implement the same or similar functions in different ways.[14]

Of clones, near-clones, and partial clones[15]

Clones and near-clones have a lesser potential for market-destructive effects than duplications of code.

---

[14] *Lotus Dev. Corp. v. Paperback Software Int'l*, 740 F. Supp. 37, 65-67 (D. Mass. 1990) (noting differences among spreadsheet programs and competitive benefits of these differences).

[15] The term "clone" is used in a more expansive manner in *A Manifesto*. One expansion is their distinction between clones, near-clones, and partial clones. A second expansion is to describe a program whose internal design is substantially identical to another program's internal design. Both internal and external program compilations are industrial compilations of applied know-how that need some protection against substantially identical copying.

Of substantially similar and substantially different program compilations

Copyright protects against any work that is substantially similar and dependently developed. Patent protects against a second comer's product which is substantially identical to the claimed invention on an element-by-element basis regardless of any dependency. The authors of *A Manifesto* think that a 'substantial identity' standard is more suitable for software than copyright's 'substantial similarity' standard.[16]

Because programs are industrial in character and they often combine pre-existing elements in incrementally new ways, a legal regime for this kind of artefact should be careful not to interfere with such incremental re-use.[17]

Features, algorithms, or subcompilations of features often migrate to domains entirely different from those in which they originated. In the process, imitators may transform the original innovations significantly in order to integrate them into new programs. An important question is whether a legal regime should prevent such migration for a shorter time than adjacent imitation, or only require royalty payment. It is desirable to ensure that even remote market participants contribute to the cost of the initial development. However, since transplanting innovations to a wholly different market often requires additional creativity and has less potential to affect the innovator's own market, such re-use in new application should be permitted sooner, or with shorter compensation periods, than those close to the innovator's market.

---

[16] The SCPA adopted a substantial similarity standard for judging infringement of chip designs. The concept underlying the SCPA may be applicable to program innovations. *A Manifesto*, Section 5.4.3.

[17] Direct protection of innovation by Kingston and Kronz considers incremental innovation. See this study, 2.4 Software Development, and 2.5 [4] Software Re-use.

Interoperating programs

In order for a program to interoperate with another program, it must reproduce the other program's compilation of interface information. Whether or not interfaces should be protected by existing legal regimes has been controversial, though interfaces are valuable and often very costly to develop and they embody considerable innovation.[18]

Interoperability can be achieved by three ways: (1) the developers' publishing of interface specifications enabling third party to construct programs that can interoperate with their programs, expecting sales increase; (2) licensing of the use of the interface codes; and (3) decompilation.

Since decompilation of an interface is such a laborious process that it necessarily entails significant delay, market failure is unlikely to occur. However, if decompilation technology improves sufficiently, it may be necessary to consider whether the use of other's internal interface information should be limited for market-preserving period of time. Since internal interfaces are industrial compilations requiring skilled effort to create, it is reasonable for a second comer to contribute for their development costs.

Add-on software

Add-on software typically enhances functions of an existing program. It modifies or supplements another program's behaviour. To modify or supplement an existing program's behaviour, an add-on program must be able to interact with the existing program. In the software industry, add-on software is very common and is widely regarded as competition enhancing. From a market-oriented perspective, the add-on programmer must have a market incentive for investing in cumulative innovation.

---

[18] *A Manifesto*, Section 5.4.4.

[3] Goals and Principles for a Market-Oriented Approach[19]

According to the authors of *A Manifesto*, there are goals and principles which can be a basis of assessing the best option among possible legal mechanisms for protecting program know-how. Although any one legal regime cannot achieve all the goals and principles, it is important to achieve a balance between conflicting principles, and to satisfy as many principles as possible. The goals and principles are the following:

(1) *Build on existing legal foundations*

Because in some respects, existing legal regimes appropriately protect software innovations without distortion of their basic principles, a total revision may be unnecessary. Copyright law, for example, has provided a simple and effective means of deterring wholesale copying of source and object code, expressive texts, pictures, or audiovisual material. Any new legal regime should supplement protection of existing legal regimes.

(2) *Focus on the most serious problems*

Since no legal regime can solve all problems and solve them perfectly, the goal should be focused on a workable solution to the most serious problems in the software industry.

(3) *Provide reasonably predictable scope and duration*

A legal regime that protects program behaviour and industrial design should be reasonably predictable as to scope and duration of protection. This will reduce the potential for litigation.

(4) *Be Responsive to the characteristics of software*

The regime should be responsive to the nature of software. The regime should provide

---

[19] Ibid., Section 6.

protection of the true sources of value in software: behaviour, the industrial designs that produce behaviour, and conceptual metaphors.

(5) *Be technically coherent*

The regime should make legal distinctions that are technically coherent.

(6) *Evolve naturally*

The regime should be able to evolve naturally as the software technology evolves.

(7) *Encourage dissemination*

The regime should encourage disclosure and dissemination of program know-how, facilitating improvements and new applications.

(8) *Encourage innovation*

The regime should encourage useful innovation and discourage overheated innovation.

(9) *Avoid market failure*

The regime should avoid market failures.

(10) *Provide reasonable lead-time*

The regime should provide reasonable lead-time.

(11) *Be attuned to the development rate*

The regime should be attuned to the rate of development in the market. The rate depends on the state and pace of innovation in the market, and the relative maturity of the market.

(12) *Provide an opportunity to recoup investment*

The regime should provide an opportunity for innovators to recoup their R&D expenses as far as their work is valuable innovation. The focus should be on stimulating investment in productive innovation. The period of protection can be determined by the assessment of the duration that would permit recoupment of efficient expenditures on

R&D.

(13) *Avoid duplications*

The regime should avoid wasteful duplicate effort. Substantial societal costs are incurred when program know-how is kept as a trade secret: the costs of maintaining secrecy, reverse engineering or recreating the know-how. While compiled disclosed know-how eliminates these wasteful costs, the vulnerability of the know-how damages the developer. As far as the know-how in software products is revealed on their face, no one will want to pay for what they can obtain for free.

(14) *Share costs*

Market participants should share R&D costs in a competition-enhancing way. A second comer may have choice between contributing to the costs of the R&D and refraining from appropriating the innovation for a market-preserving period. The period should be short enough to balance between licensing and waiting for its transferring into public domain.

(15) *Provide remuneration*

A market-oriented legal regime should recognize the value of an innovation regardless of commercial success of the product embodying the innovation. Since many valuable and incremental innovations in software appeared in commercially unsuccessful products, the regime should find a way to encourage innovation, independently of whether or not it is in successful product.

(16) *Provide incentives to agree rather than to litigate*

A market-oriented legal regime should provide incentives to agree rather than to litigate. To avoid litigation costs as well as high transaction costs of licensing, it is desirable to provide standard licensing arrangements.

(17) *Distinguish among different kinds of second comers*

A market-oriented legal regime should consider a number of factors in determining whether second comers should pay a standard fee to use an innovation or should be blocked from use for a period of time. Such factors may include the relative size of the appropriated innovation, the manner by which the second comers accessed to it, the degree of similarity, the extent of improvement, and the proximity between the markets where each of the innovator and the second comers are operating.

(18) *Be self-executing*

The more self-executing a legal regime is, the more "market-friendly" it will be. A market-oriented legal regime should minimize the costs of obtaining protection. One way to achieve both market protection and self-execution is to provide some degree of lead-time protection automatically. This is especially important in rapidly developing technology fields, such as software.[20] Software developers generally need legal protection most in the first few years after they have introduced an innovation to the market. Since patent law provides no protection to an invention until the patent actually issues, many software innovations are vulnerable to appropriation in the first years after they have been introduced into the market. When a patent is granted, the useful commercial life of the product embodying an innovation is usually over, due to the fast pace of innovation in the software industry. A legal protection for the first years of a technical innovation in software would be necessary for software industry.

(19) *Minimize barriers to entry*

A market-oriented legal regime should minimize barriers to entry. Artificial barriers to entry, which were intended to cure market failure, may cause another kind of market failure.

---

[20] *A Manifesto*, Section 6.1.18.

(20) *Promote consumer welfare*

A market-oriented legal regime should promote consumer welfare. The regime should be cautious of both overprotection and underprotection.

On the other hand, Reichman articulates the following elements on which a market-oriented framework for protecting software innovations should be built:[21]

(1) Treat industrial compilations of applied know-how as the objects of protection.

(2) Provide artificial lead-time to prevent market failure.

(3) Develop a menu of user liabilities that sensibly allocate the costs of R&D among innovators and borrowers.

(4) Allow registration with disclosure.

(5) Supply pro-competitive standard ground rules.

(6) Develop legal and organizational means to enable collective action to enforce the liability framework.[22]

[4] Frameworks for a Market-Oriented Legal Regime

A market-oriented legal regime should pursue the satisfaction of as many of the principles discussed above as possible. An ideal legal regime may protect just long enough to enable the software innovator to enjoy the same lead-time as other innovators who contributed equal value to the market.[23] The required amount of artificial lead-time would depend on the amount of natural lead-time already available, in accordance with the difficulty of reverse engineering. If such individualized tailoring were possible,

---

[21] J.H. Reichman, *Legal Hybrids Between the Patent and Copyright Paradigms*, 94 Colum. L. Rev. 2432, 2455-62 (1994).
[22] See Dreyfuss, *Expanding*, pp. 23-53. See also this study, 6.2 Compensatory Liability Regimes.
[23] *A Manifesto*, Section 7.1.

each innovator could count on the chance of earning the return to justify its investment.

[a] Automatic Blockage of Cloning

However, since such individualized tailoring would be unfeasible, a more plausible approach would be to protect program behaviour and other industrial design elements of program against cloning for a period of time sufficient to avoid market failure. Protection against cloning by law might commence from the first public marketing of the program embodying it.[24]

The advantages of this approach are the following: (1) It is low-cost and self-executing; (2) It would directly protect against the next most trivial means of acquiring functional equivalence after entire duplication of code, identical copying of program compilations and engineering designs; (3) Because it would be limited to protecting against identical or near-identical copying, it would be relatively predictable; and (4) After the duration of protection that would be consistent with lead-time, it would allow compiled know-how to be re-used thereafter, promoting cumulative innovation, competitive add-ons, and the standardization of efficient solutions.

The disadvantages are the following: (1) It seems too weak, because substantially similar implementations would not be regarded as clones; (2) Without a registration system, it may be difficult for second comers to know when the anti-cloning period expires; and (3) It will not give any compensation to the innovator whose own commercialization effort is a failure, even though whose innovation benefits the market

---

[24] This point is similar to the concept of the direct protection of innovation by Kingston and Kronz. The object of innovation patent by Kronz is the invention actually reduced to practice, and commercialized. The subject-matter of innovation warrant by Kingston is an investment which turns an idea into concrete reality. See Kingston, *Direct protection*, p. 47.

and is exploited by others with success.

[b] Automatic Anti-Cloning Protection Followed by an Automatic Royalty-Bearing Licence

The disadvantages of automatic blockage of cloning protection lead us to consider a two-phase protection regime. The first phase would block clones in order to give innovators the opportunity to develop a market niche. The second phase (automatic licence period) would require users to pay standard licensing compensation to the innovator.[25] By this second phase, regardless of the commercial success, the innovator can receive some compensation from others who use his innovation. The duration of the second period should be short under the principles discussed above.

However, without a registration system to identify and describe the subject-matter to be protected, it would be difficult to know when blockage periods ended, when the automatic licence period commenced, and what was protected. Transaction costs for licences could be low, if the law implements a standardized licensing form.

[c] SCPA- Like, Automatic Anti-Cloning Protection with Registration

Semiconductor chip design requires skilled efforts to make incremental improvements in the selection and arrangement of functional elements. As with software, semiconductor chip designs typically bear much of the incremental technical innovation on the face of the product in the market. Chips are very costly to develop, but once developed, their designs are vulnerable to fast and inexpensive appropriations. Second

---

[25] This point is also found in the compensatory liability regime. See this study, 6.2 [3] Implications of a Compensatory Liability Regime.

comers can acquire functional equivalence to an innovative chip design by copying products in the market. Due to these characteristics, semiconductor chips, like computer program, are difficult to fit into the framework of traditional IP law.

SCPA provides automatic anti-cloning protection to semiconductor designs from the date of the first commercial distribution of a chip embodying them.[26] This protection lasts for two years unless a chip developer registers the design at the Copyright Office.[27] The SCPA registration process, like that of copyrights, involves only a light examination of the application before a registration certificate issues. A timely registration will extend the duration of protection to ten years. The SCPA certificate, like that of copyright registration, constitutes *prima facie* evidence that the holder has SCPA rights. Under SCPA, others are free to use aspects of a chip compilation as long as they design their competing chips independently.

SCPA's actual subject-matter is "mask works", that is, the set of stencils or "masks" used in the manufacture of chip layers under the technology in common use when SCPA was devised.[28] A set of "mask works" for a particular semiconductor chip design must accompany the application for registration sent to the Copyright Office.[29] A registration system has worked reasonably well for SCPA because mask works are an intermediate work-product of the manufacturing process that can accompany the registration application. Registration of chip designs also remedies some defects of a pure anti-cloning approach. For instance, applications for registration must state the date

---

[26] Protecting from the date of the first commercial distribution appears to be similar to the concept of Kronz's Innovation Patent system. Protection for a mask work commences on either the date it is registered with the Copyright Office, or the date on which it is first "commercially exploited" anywhere in the world, whichever is first. See 17 U.S.C. § 904(a) (1988).

[27] 17 U.S.C. § 904(b), 908(a), 908(e) (1988).

[28] 17 U.S.C. § 902; § 901(a)(2) (defining "mask work").

[29] 17 U.S.C. § 908(c); 37 C.F.R. 211.5 (1993). The choice of mask works as the subject-matter for the SCPA protection regime has been criticized because, as chip technology evolves, masks are less frequently used, making SCPA potentially obsolete. http://www.bustpatents.com/aharonian/softcopy.htm accessed 29 January 2002.

on which commercial exploitation commenced. This helps copyists determine when legal protection ceases. Masks deposited at the Copyright Office also make it easy to know with certainty the exact design protected. Registration also makes it easier to record transfers of IPRs.[30]

However, registration of software innovations would not be easy to achieve because there is no intermediate design document uniformly prepared by software developers. Software developers would be reluctant to register a design document that disclosed all of the internal design elements of their programs, and other information that they could now protect as trade secrets because of the difficulties of gaining access to it by decompilation. So, SCPA's registration system would be unworkable as a model for software.[31]

[d] Modified SCPA Approach: Some Automatic Protection Complemented by Registration of Innovative Elements

This approach provides a period of automatic anti-cloning protection and an opportunity to register innovative software compilations or subcompilations in order to acquire a longer period of exclusivity or a period of compensation under a standard licensing arrangement. The subject-matter of this might include a new user interface design, a macro language, and a new algorithm. The applicant need not register the product as a whole, as is required under SCPA. This approach may best match the design principles.

Registration should probably be required to take place within a year or two of the first commercial distribution of a product. The legal regime should employ a copyright-like registration process (rather than a patent-like examination procedure). It allows

---

[30] 17 U.S.C. § 903 (1988).
[31] *A Manifesto*, Section 7.4.

later opportunities to challenge the qualification of the registered material for protection. This would minimize the costs gaining protection in the software industry where the pace of innovation is fast.

Registration might give an extended period of exclusivity or an automatic royalty-bearing licence[32] available on standard terms after expiration of the unregistered protection right. The latter would remove the transaction costs of licensing. Reasonable fixed fees would encourage second comers to compensate the innovator rather than duplicating effort. They would also serve for the innovator in licensing negotiations.

[e] A Market Segment Approach

Although exclusive rights regimes do not generally connect the scope or term with the market proximity of a second comer's product,[33] a market-oriented regime for protection of industrial design in software might do so. A second comer's ability to enter the market might be regulated according to how close the second comer's market is to the innovator's market.

Crafting an appropriate scope for derivative work rights in an industrial property law regime is a very difficult problem. While granting very broad rights over derivative work can unduly hamper competition, providing no means for control over derivative uses of an innovation may reduce incentives to innovators. Regulating use of innovations by market segment might provide a mechanism which could achieve the balance.

---

[32] See Reichman's Liability Regime.

[33] Trademark laws consider market segment. Goods bearing the name of a first comer's product are generally regarded as noninfringing of trademark rights if the second comer's product operates in a market remote from that of the first comer.

Some regulation of adjacent markets may be necessary, because a second comer's exploitation of an innovation in an adjacent market would have some potential to undermine the innovator's ability to use the innovation in the adjacent market. However, if the second comer's market is very distant from the innovator's market, the second comer's use of the innovation in that market may not have market-destructive effects and should be regulated lightly. Derivative uses of an innovation in remote markets might be blocked for a shorter period of time, or might be subject to an automatic licence rather than a lead-time blocking period.

The market segment might also affect the degree of similarity. When an innovation migrates to a remote market, the innovation will often need to be adapted to its new environment and the adapted innovation may not be identical to the original innovation even if it accomplishes exactly the same behaviour. Market proximity is also an appropriate factor to consider in setting standardized royalties for re-use of a registered innovation, especially if industry-wide blanket licences establish to implement a liability regime.

[f] An Improvements-Oriented Approach

An improvements-oriented approach distinguishes those who made improvements to an innovative program compilation of other's product from those who imitated the compilation without improvements. An improver might come to the market sooner than a copier. Or, the improver might license for a standardized fee, while the copier be blocked.

Consideration of improvements would be desirable if a substantial similarity

standard (rather than a substantial identity standard) were selected as the standard by which to judge whether a second comer had unfairly interfered with the market opportunities of a software innovator. However, it is often difficult to know whether differences from the original are improvements or mere attempts to avoid liability. Although consumers favour an improved version, this choice made by consumers does not exactly distinguish between substantive improvements and price improvements.

[5] Alternative Courses of Action

According to the authors of *A Manifesto*, policymakers have at least three options for legal protection for software innovations. One is doing nothing. The second is making minimal changes (i.e. Anti-Cloning Protection) to address the urgent underprotection problem: the lack of protection against cloning program behaviour and other industrial design elements of software. The third is establishing a registration-based system that would supplement anti-cloning protection to ensure that innovators would receive contributions from those who wish to reimplement their innovations.

[a] Doing Nothing

Proponents of the status quo suggest that existing legal regimes should be given additional time to evolve. However, as discussed above, existing legal regimes cannot provide appropriate protection for software innovations because existing regimes and the valuable innovations embodied in software are fundamentally mismatched. Furthermore, several factors may adversely affect the current situation: (1)

decompilation technology may improve enough to make it very easy for competitors to acquire the industrial design of program internals that can be kept secret today; and (2) a number of questionable patents for software-related inventions may impede competitive development and follow-on innovation in the software industry.

[b] The Minimal Change Option: Anti-Cloning Protection

Because of their vulnerability to trivial acquisitions of equivalence, externally perceptible compilations of applied program know-how (e.g. behaviour and user interfaces) may need some artificial lead-time that classical IP regimes do not provide. Although some natural lead-time is available to those who introduce innovative behaviour and user interfaces to the market, this lead-time is inadequate to the needs of the industry. These compilations should be protected from commercial appropriations that destroy lead-time, and from clones, near-clones, and partial clones.

Because a market-oriented legal regime for a fast-moving technology like computer software should minimize administration, protection against cloning should start automatically from the first commercial distribution, and last long enough to give innovators sufficient lead-time to develop a market niche. However, it should not last so long as to impede the incremental development of software technology or the creation of new standards in the marketplace.[34] It might be useful to adopt short terms of protection for program compilations, which have at least minimal creativity in the compilation and do not consist entirely of standard or commonplace elements arranged in a standard or commonplace way.

---

[34] *A Manifesto*, Section 8.2.

The misappropriation branch of unfair competition law[35] is a possible base on which a common form of anti-cloning protection for software could be built. To provide software developers some lead-time, the common law could evolve anti-cloning rules for software innovations, employing market-oriented principles and reflecting the primary dimensions discussed before. The software industry would match the market-oriented, lead-time approach because the industry has innovated and prospered enough.[36]

[c] Registration and an Automatic Licensing System[37]

Since anti-cloning legislation is only a partial solution, a broader solution would be necessary. It will include a registration system for innovative compilations of applied know-how embodied in software. This system would help establishing a documented prior art that could be useful to the development of software engineering. This registration system might also provide incentives for innovators to disclose innovative algorithms and other internal design elements of software, because they can get compensation for the disclosed innovations.

It is desirable to devise a legal framework that is adaptable as the software technology and markets evolve, because the evolution of technology and markets may affect the legal situation and may open a number of opportunities for electronic markets.

With technological developments, second comers may more easily acquire behavioural

---

[35] *International News Service v. Associated Press (INS)*, 248 U.S. 215 (1918). Associated Press brought suit against International News Service because International News Service was appropriating news from early editions of Associated Press-affiliated newspapers and publishing it in its own affiliated newspapers that competed directly with the Associated Press papers. To preserve incentives for Associated Press to invest in news-gathering, the US Supreme Court decided to give it lead-time protection in the commercial distribution of the news.

[36] *A Manifesto*, Section 8.2.

[37] Ibid., Section 8.3.

equivalence of an innovator's software product. Improvements in software technology may speed up the cycles of the lead-time. Advancements in decompilation technology may enable second comers to read all of the know-how embodied in program internals as easily as if they were on the surface of the product. If these happen, it is necessary to adjust the legal regime to prevent market-destructive appropriations, and to ensure adequate lead-time protection.

On the other hand, technological developments for protecting intellectual products may also have disturbing effects on the legal equilibrium as well. Such technology might make any legal rules about decompilation obsolete. In this case, the issue could be how to induce firms to disclose or license internal program know-how to avoid wasteful duplications of effort.

Developments of software technology may open a number of opportunities for electronic markets. Electronic markets for software may evolve out of electronic repositories of re-usable software components that already exist.[38] Growth of these repositories is foreseeable as digital networks expand. The electronic repositories of algorithms that already exist on the Internet tend to be public libraries of algorithms. If these repositories evolve into software exchanges, innovators who want to be compensated for other's uses of their innovations might register them on an automated licensing basis. The repository can search to determine whether or not the algorithm already exists. If not, the registration can be accepted on the condition that it becomes nullified if the representations made at registration were later found to be false. After expiration of the automated licence term, the algorithm would belong to public domain. This system would have advantages such as enhancing the public's access to algorithms; increasing the storehouse of knowledge available to software engineers; and

---

[38] See this study, 2.5 [3] Componentized Design Architectures.

providing a means for innovators to receive compensation.

[6] Subject-matter of the Protection

The authors of *A Manifesto* think that the most important properties of programs are (1) *their behaviour, i.e. the set of results brought about when program instructions are executed*, (2) *the industrial design responsible for producing behaviour*, and (3) *the conceptual metaphors that give behaviour coherence*.[39]

The primary source of value in a program is its behaviour, not its text.[40] A really important characteristic of programs is the fact that they behave. Behaviour is an essential part of programs. Program behaviour consists of all the actions that a computer can perform by executing program instructions. The authors of *A Manifesto* have tried to provide a new legal regime for the protection of "*the applied know-how found in the design of program behaviour*."

Conceptual metaphors are valuable as organizing principles for program behaviour, as well as for the virtual worlds and objects they create.[41] An innovative conceptual metaphor is one of the most valuable types of software innovation. The authors of *A Manifesto* assert that the legal regime should find a way to protect the effort that produces such valuable new tools as conceptual metaphors.

Computer programs are inherently compilations of sub-components.[42] Program construction requires "*selection and arrangement of useful components*". Software engineering involves assembling components (e.g. data, data structures, and algorithms)

---

[39] *A Manifesto*, Introduction.
[40] Ibid., Section 1.
[41] Ibid., Section 1.
[42] Ibid., Section 1. See also this study, 2.5 [3] Componentized Design Architectures.

to produce a desired behaviour. Thus, programs are "*compilations of behavioural components*", because larger programs are built from smaller programs and programs behave. Writing programs is an industrial design process similar to the design of physical machines.

On the other hand, innovation in computer programs is largely incremental and cumulative in character. It is "the product of the skilled use of know-how to solve industrial design tasks". Software engineering involves the re-use of known elements in a new and efficient manner.[43] While the incremental nature of innovation in software largely precludes patent protection, the predominantly functional nature of program behaviour and other industrial design aspect of program preclude copyright protection.[44] Moreover, trade secrecy cannot protect much of the know-how used in software design, since such know-how is largely evident in distributed products and trade secrecy cannot protect what is not secret. In sum, patents do not protect incremental innovations of software because of lacking in being an inventive step. Copyright protects only text and text is largely independent of behaviour which is the primary source of value in a software. Trade secrecy cannot protect the know-how used in software which is not secret after marketing.

In response to these problems, the authors of *A Manifesto* propose a two-part solution.[45] First, it is a protection scheme organized around the source of value in software, i.e. program behaviour and *the applied know-how that produces it*. Second, it is a protection scheme based in principles of market economics and market preservation.

According to their explanation about the characteristics of (and the source of value in) computer software, the subject-matter that they have sought to protect by the

---

[43] See this study, 2.5 [4] Software Re-use.
[44] *A Manifesto*, Section 2.
[45] Ibid., Section 2.

market-oriented legal regime are the following:

(1) Program behaviour, that is, the set of results brought about when program instructions are executed.

(2) The industrial design which is responsible for producing behaviour.

(3) The conceptual metaphors that give behaviour coherence, and/or that organize principles for program behaviour, virtual worlds and objects.

(4) Selection and arrangement of useful components.[46]

(5) Compilations of behavioural components.

(6) The applied know-how that produces program behaviour.

[7] Debates on *A Manifesto*

Derrick agrees that there are many problems with trying to fit computer software under current laws. He argues that computer software is "a different type of animal and it requires a different type of cage."[47] After discussing the problems with current copyright and patent laws in protecting computer programs, he introduces the goals and principles as well as the proposed frameworks for market-oriented legal regime of *A Manifesto* as a solution.

With regard to the failure in providing adequate implementation details, however, *A Manifesto* is criticized by Gordon and Goldstein.[48] Even though they are persuaded by *A Manifesto* and admit that the law should be amended to protect software behaviour,

---

[46] See this study, 2.5 [3] Componentized Design Architectures.

[47] Douglas C. Derrick, *It Doesn't Fit: The Dilemma of Computer Software and Patent/Copyright Law*, E Law – Murdoch University Electronic Journal of Law, Vol 3, No 1 (May 1996). www.murdoch.edu.au/elaw/issues/v3n1/derrick.html accessed 1 June 2002.

[48] Wendy J. Gordon, *Assertive Modesty: An Economics of Intangibles*, 94 Colum. L. Rev. 2579 (Dec. 1994); Paul Goldstein, *Comments on A Manifesto Concerning the Legal Protection of Computer Programs*, 94 Colum. L. Rev. 2573 (Dec. 1994), cited by Mark Aaron Paley, *A Model Software*, p. 9.

they suggest that the authors of *A Manifesto* should provide not just a proposal in itself, but explain how these principles would work in concrete form.

Arguing that *A Manifesto* does not provide a detailed *sui generis* statute implementing its "market-oriented" solution, and that *A Manifesto* simply provides a long list of goals and principles, Mark Aaron Paley suggests *A Model Software*. *A Model Software* assumes that the true sources of software value are its algorithms. It differs from *A Manifesto* primarily by defining what is protectable. While *A Manifesto* complicates protection by dividing software into five entities,[49] *A Model Software* instead uses a much broader definition of the term "algorithm" which may contain all five of the software behaviour entities, and tries to protect them with a single scheme.

Ginsburg, one of the opponents of a new legal regime, argues (1) that the computer industry is currently thriving, (2) that the copyright does, to some extent, protect "behaviour" of computer programs, and (3) that the alternative proposals are unlikely to achieve domestic enactment or broad international agreement.[50] However, Ginsburg's position (1) and (2) may be criticized as follow: (1) 'Thriving market' theory cannot justify the argument that no *sui generis* regime is necessary, because tomorrow's market could be much better if more appropriate regime could be provided today. (2) Copyright cannot protect the idea or process which is underlined in the sequence of behaviours of computer programs, since copyright protection cannot extend to any idea, procedure, process, system, or method of operation regardless of the form in which it is embodied in such work.[51]

---

[49] Program code, program compilation, subcompilations, algorithms, and features. See *A Model Software*, p. 9.

[50] Jane C. Ginsburg, *Four Reasons and a Paradox: The Manifest Superiority of Copyright Over Sui Generis Protection of Computer Software*, 94 Colum. L. Rev. 2559 (Dec. 1994). Available at www.law.cornell.edu/commentary/intelpro/gns94txt.htm accessed 1 June 2002.

[51] 17 U.S.C 102(b) (1988).

6.2 Compensatory Liability Regimes[52]

[1] Introduction

Both patent and copyright protection for software innovations are unsatisfactory due to the special characteristics of software. While patents tend to over-protect small innovations in the software industry, copyright provides under-protection resulting in too little incentive to first comer. To solve this problem, Jerome H. Reichman proposed a compensatory liability regime.

[2] Mechanics of a Compensatory Liability Regime

Reichman explains the compensatory liability regime by a hypothetical 'green tulip' problem. The proposed compensatory liability scheme obligates second comers to pay equitable compensation for borrowed improvements over a relatively short period of time.[53] First comer (Breeder A), who has developed a green tulip, is entitled to a specified period of artificial lead-time during which the use of the green tulip requires not authorization but compensation. Breeder A's entitlement operates as a liability rule and not as an exclusive property right. He does not have the right to deter second comer (Breeder B), from borrowing his innovation (the green tulip) and Breeder B need not seek Breeder A's permission to use the innovation in the green tulip as long as Breeder B is willing to pay.

---

[52] Dreyfuss, *Expanding, 2. Of Green Tulips and Legal Kudzu: Repackaging Rights in Subpatentable innovation*, pp. 23-53.
[53] Ibid., p. 39.

If Breeder B remains patient and waits until the period expires, he may use the innovation freely. An impatient Breeder B who possesses sufficient technical know-how of his own can independently develop a green tulip variety without compensation to Breeder A. Breeder C is treated like Breeder B. Breeder C is also free to use Breeder B's improved variety (a red, white and green tulip) to his other follow-on products without seeking authorization. Borrowing the red, white and green tulip will require compensation to both Breeder A and B (if it is during the liability period of Breeder A and B). If Breeder B and C do not borrow from Breeder A during the liability period, and accordingly they pay nothing to Breeder A, Breeder A will nonetheless have benefited from a period of artificial lead-time.[54]

[3] Implications of a Compensatory Liability Regime

The compensatory liability regime takes the form of an automatic licence[55] without the power to exclude. Despite the weakness of the right, a rightholder would not necessarily collect less income. An aggressive second comer's applications might yield far more income than the first comer would have obtained if he had denied the licence or granted it exclusively to a more congenial licensee. The possibility of unexpected returns arises especially when several second comers become interested in multiple follow-on applications (that could produce a cumulative benefit in excess of what the first comer's own business plan might otherwise have yielded). On the contrary, when Breeder B and C accomplished their own innovations, Breeder A must contribute to the development costs of Breeder B and C by paying compensation to them.

---

[54] Ibid., pp. 40-41.

[55] See this study, 6.1 [4] [b] Automatic Anti-Cloning Protection Followed by an Automatic Royalty-Bearing Licence.

Reichman asserts that society would be cumulatively better off under the regime, while Breeder A is not always worse off and Breeder B retains sufficient incentives to play the game. Once Breeder B opts to make contributions to Breeder A's costs, he places himself in a position to collect similar contributions from Breeder C and even from Breeder A, who will often want to exploit the second comer's follow-on innovation in order to keep up with the state of the art.

Thus, according to Reichman, the proposed compensatory liability regime eliminates the economically unjustifiable tendency of exclusive property rights to allocate ownership of follow-on applications either to the first comer (at the expense of others) or to second comers (at the expense of the initial innovator).[56] In this state, the first comers can take their business strategies knowing that second comers must pay compensation for follow-on applications of the small scale innovation in which they plan to invest, and knowing also that they themselves are entitled to borrow back any such follow-on applications in return for compensatory liability. At the same time, the second comer's legal ability to borrow freely the first comer's innovation is limited in practice by the need to consider the profitability of his contribution to the first comer's costs. Within the specified time limits, this automatic licence should empower all the players to move between the status of lenders and that of borrowers, unimpeded by artificial legal barriers.

The developments of the Internet and e-commerce have reduced the cost of copying, shrunk lead-time, and thereby increased the risk that small scale innovators will keep their know-how secret. The enactment of a general purpose innovation law on modified liability principles would lessen these risks, because it would offer innovators a way to reduce market failure. The liability regime would also provide some protection

---

[56] Dreyfuss, *Expanding*, p. 51.

for commercially valuable, small scale innovations, and thus it would be possible to restrict the dominant patent-copyright dichotomy to truly non-obviousness inventions and original works of authorship.

Reichman concludes that the modified liability rule would resolve the difficulties of property-based rules for small innovations by providing a designated period of artificial lead-time, during which firms are permitted to borrow on another's sub-patentable innovations whenever they contribute to the costs of development.[57]

[4] Subject-matter of the Compensatory Liability Regime

How to enable entrepreneurs to appropriate the fruits of their investments in cumulative and sequential innovation without impeding follow-on innovations and without creating barriers to entry has become one of the most difficult issues that law and economics of IPRs need to address.[58] Instead of the breakthrough or pioneer inventions of the past, it is the routine engineers' (1) *cumulative and sequential working out of shared or common technical trajectories* that increasingly drives the post-modern economy.[59] The routine engineers produce technical know-how: i.e. (2) *a store of information about methods or processes of production, which confers commercial advantages on those who possess it.* The production of today's cutting-edge technical know-how is vulnerable to free-riding duplicators. This vulnerability of "small grain-sized innovation" to copiers breeds fears of market failure.

By suggesting a compensatory liability regime, Reichman seeks to identify some

---

[57] Ibid., p. 52.
[58] Ibid. , p. 23.A market-oriented legal regime should minimize barriers to entry. See this study, 6.1 [3] Goals and Principles for a Market-Oriented Approach.
[59] Ibid., p. 26.

of the historical difficulties in protecting (3) *small grain-sized innovations that do not rise to the level of novel and non-obvious inventions or original and creative works of authorship*.[60] Under the regime, within a designated period of artificial lead-time, firms are permitted to borrow one another's (4) *sub-patentable innovations*, only if they contribute to the costs of development.

Reichman tries to protect the objects of the following:

(1) Cumulative and sequential working out of shared or common technical trajectories.

(2) A store of information about methods or processes of production that confers some commercial advantages on those who possess it.

(3) Small grain-sized innovations that do not rise to the level of novel and non-obvious inventions or original and creative works of authorship.

(4) Sub-patentable innovations.

According to the above list, the subject-matter of the compensatory liability regime can be defined as "sub-patentable innovations that do not rise to the level of novel and non-obvious inventions or original and creative works of authorship, but can confer commercial advantages on those who possess them."

6.3 Utility Models

[1] Introduction

The European Commission has presented a proposal for a Directive approximating the

---

[60] Ibid., pp. 23-24.

legal arrangements for the protection of inventions by utility model.[61] Though this Directive is aimed at harmonizing the main provisions of national law regulating the protection of inventions by utility model, this form of protection appears to be more suitable for inventions which have a limited degree of inventiveness (a lower level of inventiveness than that required for a patent) and a relatively short life, since it is more flexible and less burdensome than the patent.[62] Utility models can therefore be regarded as a more effective tool to SMEs than patents as far as it is concerned with the inventions which have a lower degree of inventiveness.[63]

The US has no utility model system, but patents cover inventions in this subject area.[64] There is nothing in the US law which provides patent-like protection using a lower standard than obviousness, despite the fact that it is employed by many countries.[65] Therefore, inventions which only qualify for utility models in Korea, Japan or Brazil would not be patentable in the US. This might be related with the dominant

---

[61] Commission of the European Communities, *Proposal for a European Parliament and Council Directive: approximating the legal arrangements for the protection of inventions by utility model*, 12 December 1997 (hereinafter, *Proposal for utility model*). This is available at www.patent.gov.uk/about/press/releases/1998/europarl.pdf accessed 10 June 2002; *Amended proposal for a Directive on the protection of inventions by utility model*, Commission of the European Communities, 7 December 1999 (hereinafter, *Amended Proposal*). Compared with the *Proposal for utility model*, the *Amended proposal* have the following features: (1) the field of application can cover *processes* and *computer programs* as well as products, but exclude biological material and chemical or pharmaceutical products and processes, and (2) in order to reinforce legal certainty and the rights of third parties, third parties as well can request a search report.

[62] http://europa.eu.int/comm/internal_market/en/intprop/indprop/utility.htm accessed 25 June 2001.

[63] Utility models are considered particularly suited for SMEs that make minor improvements to existing products. www.1000ventures.com/bu.../sme_guide_utility_models_bywipo.htm accessed 6 June 2002.

[64] Though the US has no such system of protection, it has design patent law. The design patent law is part of the patent statute, 35 U.S.C. 171, which provides that "Whoever invents any new, original and ornamental design for an article of manufacture may obtain a patent therefore, subject to the conditions and requirements of this title." http://classes.lls.edu/spring2002/internationaltech-mcdermott/handout2.doc accessed 7 June 2002. John Giust, *Comparative Analysis of the United States Patent Law and the New Industrial Property Code of Brazil*, Hastings International and Comparative Law Review, Spring, 1998, 21 Hastings Int'l & Comp. L. Rev. 597.

[65] Countries currently employing a utility model system include Korea, Japan, Germany, France, Italy, Spain, Austria, Denmark, Belgium, Portugal, Brazil, Poland, Mexico, the Philippines, Uruguay, Taiwan, Australia, Chile, Morocco, OAPI member countries in Africa, China, Greece, Finland, Malaysia, Guatemala, Indonesia, Russia, Ukraine, and Estonia. The United Kingdom, Luxembourg and Sweden have decided to do without utility model protection. See *Proposal for utility model*.

status of the US in the technology development in comparison with the other countries.[66] In this respect, it is interesting to note that Japan has gradually changed its attitude toward utility models with technology development.[67] For example, the rapid development of technology brought a theory favouring the abolition of the utility model law (*Theory of Abolition*).[68]

[2] Mechanics of the Utility Model Regime

A utility model is a registered industrial property right which confers exclusive protection for a technical invention.[69] It largely resembles a patent in that the invention must be 'novel', 'inventive' and capable of industrial application, though generally the level of inventiveness required is not as high as it is in the case of patents. The main features of the utility model compared with a patent are a lower level of inventiveness[70] than that required for a patent, the absence of a prior examination of the protection conditions, and a limited protection period of no more than ten years.

To distinguish inventions protected by the patent system, it is necessary to define an inventive activity, which constitutes subject-matter of the utility model. According to the *Proposal for utility model*, "utility model" means *the registered right which confers exclusive protection for 'technical inventions.'* In the Member States (except for the UK, Luxembourg and Sweden) it is conferred by a variety of names: "utility model", "utility

---

[66] In the technology trade, the US was remarkably in the black while Japan was in the red in the 1986-1995. See this study, 4.1 [1] [c] The Pro-patent Policy in Japan.

[67] The number of utility model applications exceeded that of patent applications in 1906. This trend continued for more than 70 years thereafter until 1981. The number of utility model application was over 200,000 in 1987, but less than 100,000 in 1992.

[68] See this study, 6.3 [3] The Utility Model System in Japan.

[69] *Proposal for utility model*, Explanatory Memorandum: Introduction, p. 3.

[70] It is, however, very difficult to determine the difference between the level of inventiveness required for a patent and that for a utility model. In practice, it is usually determined by the application i.e. patent or utility model application, and the examiner's decision.

certificate", "sixyear patent", "short-term patent", "petty patent", "utility model certificate", etc.

Under the utility model regime, an invention would be considered as involving an inventive step if it exhibits either particular effectiveness in terms of ease of application, or a practical (or industrial) advantage. It is required that an invention should not be derived in a very obvious way from the state of the art. Examples include the following: an invention making it possible to solve a technical problem; an invention relating to the effectiveness of the use of a product in that it increases the product's usefulness by making it more effective and easier to use.

Since the utility model would be granted without prior examination of the basic conditions, i.e. novelty and inventiveness, it could be provided rapidly and cheaply, but the protection conferred is less secure. Due to this deficiency of prior examination, in order to reinforce legal certainty and the rights of third parties, it is required to have a search report in the event of either legal proceedings to enforce the rights conferred by the utility model, or extension of the protection after the initial six-year period.[71] The search report is to be drawn up at the request of either the applicant or third parties.

The suggested period of protection is a maximum of ten years, comprising an initial period of six years followed by two periods of two years, where appropriate.

The utility model confers on its proprietor the exclusive right to prevent third parties without consent from making, using, offering for sale, selling, or importing for these purposes the registered product (or the product obtained by the registered process).

The same invention may form the subject-matter, simultaneously or successively, of a patent and a utility model. In order to avoid this duel protection, a utility model (which has been granted) should be regarded to be ineffective when a patent relating to

---

[71] See *Amended proposal.*

the same invention has been granted.

[3] The Utility Model System in Japan

The creation of a technical idea is protected under two separate laws in Japan, i.e. the patent law and the utility model law. The patent law and the utility model law in Japan had very similar legal regimes and were in general closely related. However, a non-examination registration system made the two legal systems very different.[72]

The technical level of Japanese inventions was low and related to the improvement of basic technologies introduced from abroad. If patent applications filed by Japanese companies were examined by the standards applicable to patent applications filed by foreigners, the Japanese applications would most likely fail. From an industrial viewpoint, the necessity was raised to establish a utility model system which would actively protect and promote petty inventions. In 1905, in response to this the Japanese adopted the utility model law.

In the Utility Model Law of 1905, novelty was recognized with respect to a device that is not publicly known in Japan.[73] The number of utility model applications exceeded that of patent applications in 1906, (patent applications totalling 4,509 and utility model applications totalling 7,952).[74] This trend continued for more than 70 years thereafter until 1981. The number of utility model application exceeded 200,000 in 1987, but rapidly declined later to total less than 100,000 in 1992 along with the development of Japan's technology.

---

[72] JPO, Asia-Pacific Industrial Property Center, JIII, Ken-ichi KUMAGAI, Faculty of Law, Kyusyu University, *Outline of Utility Model System*, 1999, (hereinafter, 'Outline, JPO'). www.apic.jiii.or.jp/facility/text/1-03 accessed 22 September 2001.
[73] Ibid., p. 6.
[74] Ibid., p. 12.

As the life cycles of products increasingly shortened, there was a need to protect technology with a short life cycle at an early stage. To protect inventions more properly, it was proposed that the examination period should be shortened.

Non-examination system

The problem of the conventional utility model system having the substantive examination was that adequate protection could not be ensured until one and a half year after the filing of an application. In order to ensure adequate protection of technology with a short life cycle at an early stage, the substantive examination system was abolished and a non-examination system was introduced. This allows a utility model registration to be granted quickly with an examination conducted only as to basic requirements. An application which cannot satisfy the basic requirements is not granted a registration. The following cases do not satisfy the basic requirement:

- A device does not relate to the shape, construction or a combination of articles.
- A device violates the unity-in-application principle or requirements for claim description.

Since non-examination system could protect technology with a short life cycle at an early stage, it was recommended to shorten the term of a utility model right which was 10 years. The term of a utility model right was set at six years from the date of an application.[75]

Registrability report

A registrability report provides an objective evaluation of the validity of a utility model registration. It is prepared by an examiner based on prior art search for each claim. To

[75] Ibid., p. 42.

prevent a third party from incurring unexpected damages from the abuse of a utility model right granted without a substantive examination, it is mandated that the owner of a utility model cannot execute his right before he warns with a registrability report.

Suspension of trial proceeding and penalty

Since a utility model right is granted without a substantive examination, an infringement suit proceeding with the assumption of the validity of a utility model right can place the defendant in a very unfavourable position. Therefore, it is stipulated that a defendant is entitled to demand the suspension of court proceedings and the court proceedings must be suspended, in principle, on a demand for suspension.

The utility model law stipulates lighter penalties for infringement offences compared with the patent law.

Examination guidelines for utility model registration

Because the subject-matter of protection under the present law is a device relating to the "shape, construction and a combination of articles", it is not appropriate to register devices falling into the category of methods, devices of constituents and devices of chemical substances, articles not having a certain shape, animal species, and plant species.[76]

The subject-matter of evaluation for a registrability report is a device described in claims for a utility model registration. Therefore, the subject of a search should be a device described in claims. A search is conducted with respect to a device described in all claims. A search of prior art to prepare a registrability report is conducted in a

---

[76] *The Guidelines for Examination of Basic Requirements for Utility Model Registration* (1993), (hereinafter, 'The Guidelines of Utility 1993').

manner similar to that conducted in the patent examination. It is judged on this registrability report whether an application is registrable or not.

A registrability report presents materials on which parties concerned can judge the registrability of a device objectively in relation to prior art. In preparing a registrability report which denies or confirms the registrability of a device, evaluation should be made in a manner which is applicable to make a final decision in the patent examination. The evaluation of lack of novelty and inventiveness needs to be indicated for each claim.

Protection of "process"

This is a problem concerning the adequacy of a process described in a claim for a utility model registration. It is clear that a manufacturing process cannot be protected under the utility model law regardless of how good it may be.

However, devices are often found registered by manufacturing methods or steps described in claims for a utility model registration. In such a case, it is questionable how the substance of a device should be interpreted. Since the subject-matter of the utility model law is essentially a device relating to the shape of articles, a method (process) for realizing it is not a matter indispensable for the construction of the device.

Therefore, it is widely accepted that a method described in a claim should be interpreted as indirectly describing a method of attaining a certain shape, a result of the method executed. In practice, it is allowed to include the description of a method in a claim for a utility model registration. This is allowed only for convenience. A method described as such is not admitted.[77]

Another problem arises from the technical scope of a device with a method described in a claim. A method described in a claim should not be interpreted as a basis

[77] Outline, JPO, p. 30-32.

to limit or expand it. A method-related description in a claim for a utility model registration is simply a matter of expression to distinguish a new shape from a conventional shape. A unique effect that cannot be achieved from other shapes is recognized to be novel and inventive irrespective of its manufacturing process. Effects attained by a method (not attained by a shape) should not be considered.

Pros and cons for a utility model law

There has been a basic question as to whether the utility model law should be maintained or revised. Along with the rapid development of industrial technology, the *Theory of Abolition* was advocated. Advocates for the abolition asserted as follows:

- The utility model law has already completed its mission and has become obsolete.
- Protecting petty inventions only encourages technology which is not internationally competitive. The increase of utility model registrations only harms the industrial development.
- Applications by large companies exceed those filed by SMEs. However, they inevitably do so only to defend themselves from SMEs.
- The presence of the utility model system has caused a delay in the examination of more important patent applications and has almost paralysed the functions of the patent system.

Those favouring the maintenance were largely from SMEs. They argued as follows:[78]

- The utility model law is a law which still remains significant and indispensable for the protection and development of businesses, particularly SMEs.

---

[78] Ibid., p. 34.

- Since a petty invention is protected under the patent laws in other countries, denying a petty invention will lead to weaken the international competitiveness.
- If the utility model law were abolished, utility model applications would be filed as patent applications, with the result of the same burden of examination.

There was also a view favouring the modification of the utility model law, on the ground that the protection of petty inventions was still required. They asserted that protection to petty inventions has been too strong and application procedures were too strict. Measures to lessen the protection include shortening the term of the right, not granting a right to seek an injunction, and stipulating no criminal charges.

[4] Implications of the Utility Model Regime

Quick and simple registration enables the applicant to be protected within a short period of time against copies and imitations, thereby consolidating the competitive position of business, in particular SMEs.[79] Rapid registration gives temporary protection and may lead to rapid commercialization of the invention.[80] This would be useful for the protection of computer programs whose lifecycle is very short.

In the case of legal proceedings or extension of the protection after the initial six-year period, the requirement of the search report forces the proprietor to avoid excessive claims for their rights, or to abandon their unnecessary rights. Through the search report,

---

[79] Utility model can be useful to SMEs, which account for more than 99% of all European firms, 66% of all jobs and 65% of turnover in the European Community. According to the study carried out by ESRC, *Intellectual Property and the Small and Medium Enterprise*, 96.7% of all businesses in the UK have turnovers of under £ 1million. See http://info.sm.umist.ac.uk/esrcip/Projects/L5253004/final_report.htm accessed 8 October 2001. Z.A Silberston suggests that the introduction of a wider adoption of petty patents (utility models) would be the most likely to occur in the foreseeable future. See William Kingston (ed), *Direct Protection of Innovation*, 1987 Kluwer Academic Publishers, (hereinafter, Kingston, *Direct protection*), p. 213.

[80] *Proposal for utility model*, pp. 11, 12.

if the utility model is recognized as not having novelty or inventive step, the right is invalidated. Moreover, the right holders themselves suppress their excessive desire to invoke the power of law, because they are afraid that their right might be invalidated, and because they do not want to pay any unnecessary fee of the search report.

The lower novelty and inventive step requirements of utility models provide flexible conditions for obtaining protection for small technological advances such as software innovations.[81] These flexible conditions encourage companies, especially SMEs, to apply for utility model protection. Since utility models are granted without any preliminary examination to establish novelty and inventive step, they are cheaper to obtain than patents.[82] Due to their limited resources, SMEs' R&D activities often result in technical inventions involving a small inventive step, which do not necessarily satisfy the requirements for patent protection. These inventions often amount to technical improvements which, if accumulated, are as important as inventions. According to the studies carried out on the basis of utility model applications, the utility model is used in a number of industrial sectors where there is a permanent need for innovation, especially in the form of minor technical inventions.[83]

From the competitiveness point of view, due to its speed and simplicity, the utility model may help SMEs to improve their market position and to facilitate the commercial exploitation of technical inventions.[84] Business people recognize that they

---

[81] It is argued that utility model protection would provide coverage for a large area of innovations which fall between design and utility patents. See www.ipmall.fplc.edu/hosted_resources/jorda_08_27_99.htm accessed 7 June 2002. WIPO-UNITAR Academy, New York City, August 26-27, 1999.

[82] Cost is a decisive factor in the case of SMEs which tend to have limited resources and information on markets to prospect the sales of new products. Cost is also important in the case of inventions whose commercial success is uncertain. The interest shown by SMEs is primarily due to the savings of cost, time and administration. See *Proposal for utility model*, p. 16.

[83] EPO, Vienna Suboffice, position at 8 January 1993, and survey of firms in Denmark, *AIPPI Yearbook* 1986, 14.

[84] Those favouring the maintenance of the utility model law were mainly from SMEs. According to the survey by the Ifo Institute, the vast majority of firms (irrespective of their size) and independent inventors thought that the utility model could help them to establish an improved market position. See Outline, JPO,

can hold on to a competitive lead only if they can prevent their competitors from copying or imitating them for a certain period of time through effective protection measures such as the utility model. They want to show originality and to distinguish themselves from their competitors, so that customers develop a positive image of their technological capability. Firms must constantly improve their products if they are to keep or increase market shares. SMEs, unlike large firms, must step up their inventive activities if they are to face up to the stiffer competition.[85]

On the other hand, according to a study carried out by ESRC, *Intellectual Property and the Small and Medium Enterprise*,[86] SMEs preferred informal protection methods which were perceived as cheaper, more familiar and, for the most part, successful.[87] In contrast, they viewed formal legal rights, particularly those requiring registration, as expensive, time-consuming, complex and of limited value. Registered rights were less commonly reported than other legal methods of protection. The results of the study showed that SMEs tend to use formal rights only in very specific circumstances, e.g. (1) where high commercial benefits are expected; (2) where SMEs believe formal rights are likely to offer better protection than informal methods; and (3) where SMEs possess the necessary resources and the desire to acquire, maintain and enforce formal rights. Moreover, most SMEs reported no intention to pursue legal action, even when success was anticipated. The costs associated with taking legal action (money, time, difficulty of establishing infringement and risk to the reputation of the business) were felt to be prohibitive. Most SMEs wanted to allocate resources to product or process innovation, rather than acquiring and/or enforcing formal IPRs. Thus, utility model regimes, which

---

p. 34; *Proposal for utility model*, p. 17.

[85] See *Proposal for utility model*, p. 18.

[86] http://info.sm.umist.ac.uk/esrcip/Projects/L5253004/final_report.htm accessed 8 October 2001.

[87] Ibid. Formal protection practices involve the creation of legal rights and sanctions for their infringement. Informal practices attempt to restrict the necessity to enforce IPRs through legal means.

require registration, may be less appropriate than informal methods to SMEs. In addition, there is no evidence that SMEs have been their main users. Large companies as well would have seen and will see advantages in using utility model protection.[88] Utility model protection may be effective and cheap method to hinder competitors from protecting their new ideas by establishing 'utility model portfolio' which surrounds a main patent. This is particularly attractive to the large company with great resources to apply for a large number of utility models rather than to SMEs with limited resources.

Encouraging filing applications for utility models can be dangerous to SMEs.[89] The owner of the unexamined utility model must be very careful in his use, because claiming this right may cause claims for damages against the owner by the alleged infringer. Without professional advice, SMEs generally have problems in understanding claims and assessing the proper scope of claims. This may lead SMEs to abuse their rights. Without exact understanding the scope of their rights and the state of prior arts, SMEs cannot use their utility models properly. This could be bad not only for SMEs but also for the market. In these respects, there are difficulties with utility models because they are not solely SMEs-frieldly, nor without danger.

The TRIPs Agreement generally requires the strengthening of IPRs regime in developing countries.[90] The strengthening may limit the access of technology by the firms and slow down the pace of the development of technology in these countries. The TRIPs will lead to a substantial increase in flow of royalties from developing countries

---

[88] Leith P, *Software Utility Models and SMEs*, 2000 (2) *The Journal of Information, Law and Technology (JILT)*. http://elj.warwick.ac.uk/jilt/00-2/leith.html accessed 3 June 2002.

[89] Ibid.

[90] For example, with the adoption of the TRIPs Agreement, around 50 countries, which had not previously conferred product patents to pharmaceuticals, were forced to do so. See Einarsson, *TRIPS* www.grain.org/docs/sida-trips-2001-en.pdf; International Intellectual Property Training Institute (Korea), *WIPO Asian Regional Seminar on the Implications of the TRIPs Agreement for Enterprises*, December 1996, p. 139.

to developed countries.[91] The optimal IPR strategy for developing countries would likely be the same one that developed countries used during their industrial development process. This would be the case with the utility model system in Japan during its development process. This suggests how utility models could be used in developing countries.[92] The TRIPs Agreement does not set standards on utility models and breeders' rights.[93] This means that countries, in implementing national laws on utility models and breeders' rights, are not bound by any of its provisions. Utility models may have special meaning for developing countries, since utility model rights protect small innovations that prevail in the innovative process in such countries.[94] Technological developments which may qualify as inventions are relatively rare in developing countries. Patents are granted much more to foreign companies than to domestic companies.[95]

In Korea, Japan and Taiwan, the relatively weak IPR protection and the availability of second-tier IPRs like utility models and industrial designs encouraged technological learning. The second-tier systems encouraged minor adaptations and improvements by

---

[91] The process of drafting the TRIPs can hardly be regarded as a fair process. The developing countries made considerable concessions in agreeing to the higher levels of protection of IPRs demanded by developed countries. See www.southcentre.org/publications/trips/tripsmaintexttrans-01.htm accessed 5 June 2002. In addition, because the majority of IP is being created in the industrialized countries, the TRIPs shifted the global rules in favour of those countries. Developing countries went along with the TRIPs with the hope of additional access to agricultural or apparel markets in developed countries and additional technology transfer and innovation. However, Einarsson concludes that the imposing a minimum global standard of IPR protection by the TRIPs would not bring any substantial benefits for developing countries; *Intellectual Property: Balancing Incentives with Competitive Access* www.worldbank.org/propects/gep2002/chapter5.pdf accessed 6 June 2002.

[92] See Peter Einarsson and Marie Bystrom, *TRIPS: Consequences for developing countries Implications for Swedish developing cooperation*, Consultancy Report to the Swedish International Development Cooperation Agency, August 2001, see www.grain.org/docs/sida-trips-2001-en.pdf accessed 10 June 2002.

[93] The absence of these two categories may be explained by the lack of interest of the industrialized countries in these fields. See www.southcentre.org/publications/trips/tripsmaintexttrans-01.htm.

[94] Maskus argued that for developing countries to implement TRIPs in such a way to maximize the economic benefits, it would be important to develop utility models and industrial design mechanisms to promote small-scale innovations. See *Intellectual Property Rights and Economic Development: An Agenda for the World Bank Group* www.worldbank.org/html/fpd/technet/sem-sums/march5.htm accessed 6 June 2002. p. 2.

[95] www.southcentre.org/publications/trips/tripsmaintexttrans-01.htm.

local firms. Japanese IPR system has exploited utility models to encourage minor improvements over the imported machinery or equipment by domestic inventors. The utility models have allowed Japanese firms to receive protection on technologies that were only slightly modified from the original invention. Quantitative studies have confirmed that the weaker patent system employed by Japan has facilitated absorption, transfer and diffusion of technology and, contributed to the productivity growth during the period 1960-93. In many cases, the protection of utility model has improved productivity in developing countries.[96] Experience in Korea, Japan and Taiwan suggests that developing countries should distinguish between different types of patent grants, and that utility models are useful for the protection of incremental innovations.

Reichman proposes that a liability regime would be better because it would guarantee a return on subpatentable innovations. According to Reichman, utility models and industrial design systems have become less suitable for developing countries than they were before, because these systems have gradually become more proprietary.[97]

---

[96] There are more examples that utility models improved productivity. In Brazil, utility models helped domestic producers gain a significant share of the farm machinery market by encouraging adaptation of foreign technologies to local conditions. The Japanese patent system (JPS) affected Japanese technology development process. The JPS in 1960-1993 was designed to encourage incremental innovation and diffusion of technology. It encouraged a large number of utility model applications. Utility models had a strongly positive influence on total factor productivity over the period. See Keith E. Maskus and C. McDaniel, *Impact of the Japanese Patent System on Productivity Growth*, 1999 Japan and the World Economy 11: 557-74, cited at *Intellectual Property: Balancing Incentives with Competitive Access*, p. 134. Available at www.worldbank.org/prospects/gep2002/chapter5.pdf accessed 6 June 2002; Keith E. Maskus, *Intellectual Property Rights and Economic Development*, this is available at www.colorado.edu/Economics/mcguire/workingpapers/cwrurev.doc accessed 9 June 2002; www.iprcommission.org/documents/CONF_BOOK3.pdf accessed 4 June 2002. The Royal Society, Commission in Intellectual Property Rights, *How Intellectual Property Rights Could Work Better For Developing Countries and Poor People*, 21-22 February 2002.

[97] If utility models are dealt as less proprietary rights, however, they could be an effective means of encouraging domestic enterprises to undertake minor adaptive innovations for developing countries. This is because utility models are almost exclusively granted to domestic residents. See Carlos A. Primo Braga, Carsten Fink and Claudia Piz Sepulveda, *Intellectual Property Rights and Economic Development*, Technet Working Paper, www.vita.org/technet/iprs accessed 4 June 2002. In both Germany and Japan, they proved to be an effective way of allowing residents to take part in the patent system and created an incentive for the commercialization of follow-on inventions. See B. Zorina Khan, *Intellectual Property and Economic Development: Lessons from American and European History*, Commission on Intellectual Property Rights, Department of Economics and National Bureau of Economic Research, www.iprcommission.org/documents/Khan_study.pdf accessed 6 June 2002.

The Consultations on the impact of the Community utility model[98] identified the main disadvantages of protection by utility models as followed: (1) too much legal uncertainty resulting in major costs, (2) risk of the proliferation of unexamined rights, and (3) negative impact on the whole system of patents in Europe.[99] Legal uncertainty, resulting from the lack of substantial examination, could be harmful to SMEs. When they make improvements to a product or a process, SMEs could be hindered by utility models registered without examination. This obstructs the natural process of follow-on innovation and might result in costly litigation. The utility models would lower the threshold of protection and institute a parallel system that is cheaper but poorer. This could lower the standards of protection in Europe. The free-flow of unexamined utility models could undermine the value of patents. Such protection could act as a barrier to innovation since firms would not wish to invest in fields where the protection was unclear.[100]

In the 1995 consultation, about a third of the replies to the Green Paper were in favour of a Regulation setting up a Community utility model.[101] The majority of the replies rejected this possibility because a single right would be too costly,[102] and because a single right would not correspond to the real needs of industry, particularly SMEs. Protection by utility model is rarely sought in more than 3 to 5 Member States and never in the whole EU. This is mainly due to the difficulties existing in the way of cross-border applications. It would be very difficult for applicants, SMEs in particular,

---

[98] European Commission, *Consultations on the impact of the Community utility model in order to update the Green Paper on the Protection of Utility Models in the Single Market* (COM(95)370 final), Brussels, 26 July 2001.

[99] However, none of the disadvantages described above has been observed in the Member States where utility model protection is in place. See *Consultations on the impact of the Community utility model*, p. 6.

[100] With regard to this point, Khan argues that utility models were subject to abuse, but clearly the potential harm was lower than in the case of patents because of their short life. See Khan, *Intellectual Property and Economic Development: Lessons from American and European History.*

[101] See *Consultations on the impact of the Community utility model*, p. 6.

[102] The cost of translation into different official languages would be a great barrier.

to overcome the administrative hurdles caused by the different laws between Community countries. Procedure, e.g. application procedure differs in Member countries. The extent of protection as well as the term of protection varies considerably. An invention which qualifies for protection in one country may not qualify in another. This is because of different criterion of inventive step and novelty. Greece, Italy and Spain accept a lower inventive step, while Belgium and France require the same inventive step as for a patent. In addition, the lower inventive step may be interpreted in different ways. The novelty criterion as well is not the same in all Member countries. In Spain, novelty is determined by the domestic state of the art, while in the others the criterion is that of the international state of the art. These differences could not be easily harmonized, even though European Commission made the *Proposal for utility model* on 12 December 1997 (and *Amended proposal* on 12 July 1999) in order to approximate the legal arrangements for the protection of inventions by utility model.

[5] Subject-matter of the Utility Models

According to the utility model laws of Japan and the Guidelines of Utility 1993, the subject-matter of the protection in the utility model system can be defined as "industrially applicable devices relating to the shape, construction or a combination of articles, which are the creation of technical ideas by which a natural law is utilized". Devices falling into the category of methods, devices of constituents and devices of chemical substances, articles not having a certain shape, animal species, and plant species are excluded from the registration of utility model.[103]

Computer programs, which have not a certain shape, appear to be excluded from

[103] The Guidelines of Utility 1993.

the subject-matter of utility model. However, in the sense that programs are machines, programs can be regarded as a device having virtual shape.[104] In addition, the abandonment of preliminary examination to establish novelty or inventiveness, which results in simplicity and low cost, is worth considering for the protection of small technological advances with a relatively short lifetime such as computer programs.

The European Parliament proposed extending the scope of the *Proposal for utility model* to cover computer programs.[105] This was accepted by the Commission and appeared in the *Amended proposal.*[106] However, Philip Leith concludes that it is a mistake to allow utility model protection for software invention and to encourage SMEs to look for protection to this kind of device, because more and easier protection may retard European innovations, particularly in the SMEs, and because a community of experts (who have the ability to ensure the balance between right holder and the public) is not developed yet.[107] There are substantial arguments that software should also be excepted from those areas receiving utility model protection.

6.4 Direct Protection of Innovation

[1] Introduction

All the early grants of monopolies in exchange for doing something new were grants of patents for innovation, not for invention.[108] In exchange for sole rights, the patentee

---

[104] See this study, 5.2 Characteristics of Software.
[105] Leith P, *Software Utility Models and SMEs*, 2000 (2) The Journal of Information, Law and Technology (JILT). Available at http://elj.warwick.ac.uk/jilt/00-2/leith.html accessed 3 June 2002.
[106] *Amended proposal* includes inventions involving computer programs.
[107] Leith, *Software Utility Models and SMEs*, 5. Conclusion.
[108] Kingston, *Direct protection*, p. 2.

introduced a manufacture which was new to the country.

'Patents granted today related only to information, to the information which was not embodied. The means of instructing the public about the new has been replaced by a description on paper, the patent specification. Since any protection by a patent to innovation is now at one remove, how much protection an innovation receives depends upon how close is the identity between the idea (of invention) and its realization.[109] If the idea is capable of only one unique embodiment, indirect protection is as good as direct protection. If it can be embodied in several ways, however, the link between invention and innovation becomes weak, and indirect protection of the innovation becomes worthless.

In these ways, patent system has become less effective in protecting innovation. Moreover, the inventive step requirement of the patent system made it difficult for incremental innovation to be protected. The inventive step requirement effectively removed much incremental innovation from the scope of patent protection. The characteristics of this type of innovation is that once it has been done, reconstructing it from elements of prior art is very easy. Since incremental innovation emerges naturally and logically from what has been done before, it is particularly vulnerable to the patent examiner's typical examination of inventive step. Thus, adoption of the inventive step criterion meant the abandonment of patent protection for many incremental innovations.

William Kingston and Hermann Kronz have tried to extend the exploitation of the principle of patenting by reviving the direct protection of innovation. According to them, direct protection of innovation has many advantages; it may give protection to incremental innovations; it offers different protection for investment of different risks; it provides secure protection to SMEs; it makes innovation more profitable; and it

---

[109] Ibid., p. 3.

generates great increase in investment.[110] On this ground, they proposed 'innovation patent' and 'innovation warrant', respectively.

[2] Kronz's Innovation Patent

In the Kronz system, a concept or technical teaching is not protected. A concept can be protected through every possible individual embodiment of the concept. While the patent system gives a reward for ideas, an innovation patent gives a reward for turning ideas into concrete realities, i.e. for innovative action. Kronz argues that since an innovation patent refers directly to the innovative object, it offers better protection of the risky investment and must be an improved means of promoting innovation. After finding many drawbacks in the existing patent system, Kronz became convinced that it is necessary to re-establish the original doctrine of patent protection.

Features of innovation patent

The following are the main features of the Kronz proposals:[111]

1. The object of protection is not an invention but an innovation, i.e. the invention actually reduced to practice, and commercialized.
2. Anything which can be embodied in marketable new things can be protected, not just technology.
3. Processes can be protected not directly, but through the physical components involved in them.
4. Capacity to commercialize an innovation as well as technical capacity to realize

---

[110] Ibid., pp. 92-100.
[111] Ibid., p. 36.

it should be a condition for receiving protection. If either is lacking, it can be provided with a "substitute innovator" through contractual arrangements.

5. Protection grants a monopoly to make, use and sell the innovation for a prescribed period, in the same form as in the classical patent system.
6. The territorial extent of protection can be a country, a region of a country, or a group of countries by agreement.
7. The protection period would vary from case to case. It depends on the innovating firm, the market and the project.
8. Protection does not apply to the diffusion phase, just as it does not apply to the invention phase.
9. The scope of protection is defined by claims.
10. Novelty is destroyed only by "public prior use", which is established by first commercial use. It would relate only to the availability of the actual commercial embodiment to the public. Novelty is not influenced by the accessibility to any concepts or technical teaching, as long as the embodiment of the concepts or technical teaching does not exist in a fully commercial context.
11. The system would either replace or supplement the classical patent system.
12. Grants are incontestable unless the application involves fraud.
13. There is no obligation to continue use after the first act of commercialization, but this can result in substantial loss of rights.

The Subject-matter of innovation protection[112]

In the Kronz system, what can be protected is an artefact whose use is new within the jurisdiction in its commercial form. Origination of a concept, discovery, design, models

[112] Ibid., p. 37.

or prototypes do not qualify for the protection. Innovation patent is granted only to the combination of a tangible object and the initial act of commercializing it. If an innovation patent is granted, the object of innovation should be in the stage ready for commercialisation.[113] An offer to sell would constitute such an act, even though a contract is not completed yet. The only question is whether or not the object has been brought into public use for the first time in the jurisdiction by the applicant. The innovation patent is concerned only with objects (including processes) of a commercial character. What is demanded is that, before the patent is granted, the object of innovation is in the stage ready for commercialization. The innovation patent is granted, not to the first inventor, nor to the first applicant for a patent, but only to the first innovator.

Innovation patents protect entire articles, whether they contain many different "inventions" or other concepts. The entity to be protected is the article (product or process) as offered for sale or other commercial use. The principle of "unity of invention" in the classical patent system is replaced by the "unity of the goods".

The Kronz system grants protection to many incremental innovations, which could not be protected by classical patent system due to their lacking an "inventive step". Such cases might include *transposition, application, identification, formulation, selection, simplification, combination* and *aggregation.*

Transposition can be found where a technical solution to a problem in one area of technology is also shown to be applicable in another. *Application* occurs where one technology is used for another. *Identification* consists in discovery of a problem to be solved, rather than in the solution found to it. *Formulation* is to express an operating relationship between specified elements into a rule which can have a wider applicability.

---

[113] Ibid., pp. 262-268.

Selection means choosing the appropriate components out of a large number of possible ones.

The protection by Kronz system extends beyond the individual object that is actually sold in two ways: Firstly, copying it merely by substituting "technical equivalents" is banned. Simply changing components, material, scale, form, proportions or arrangements for embodying the innovation would be within the scope of an innovation patent. Secondly, the patentee is allowed to list variants of his innovation other than the one he has actually used in the market. The protection he will receive for these will not be as good as for the one he has actually adopted. Others will be allowed to make and market them if they pay a royalty.[114] Thus, the obligation to commercialize in the Kronz system forces an innovator to select out of all possible variants of his ideas to turn into concrete reality, the one which will best meet the market's needs.[115]

On the other hand, the "initial commercial act" would be defined by statute. It might include sales promotion, showing at exhibitions, commissioning plant with a view to production, supply to distributors and offer to sell.[116] Internal use, which takes place within a commercial firm, qualifies because this is considered to have consequences in the commercial world outside. Using within a public research laboratory, however, would not qualify for protection.

Direct protection by the innovation patent is similar to copyright.[117] In copyright, protection is not given to any idea or concept for the work which an author or an artist might have in mind. Copyright protects the work *itself*. There is nothing in copyright system that can be compared with "the inventive step" or "novelty" criteria of the patent

---

[114] See this study, 6.2 Compensatory Liability Regimes by Reichman.
[115] Kingston, *Direct protection*, p. 39.
[116] Ibid., p. 40.
[117] Ibid., p. 39.

system.[118] Important thing is whether or not something concrete has been produced through original effort.

Since only tangible objects can receive innovation protection, a process as such, which is not a tangible object, cannot qualify. In the Kronz system, this results in protection of a process through its components. A listing of both the hardware and the software involved in a process results in a description of the process. In a process, the innovative act only takes place when the process works.[119]

A similar approach allows innovation protection to cover computer programs and methods of doing business.[120] According to this system, a description of a method of doing business may consist of both the hardware and the software involved in the method of doing business. According to Kronz's explanation, a process controlled by computer programs can be described as an aggregate of tangible objects, interacting together. In the case of a chemical process, the apparatus used would be described, as well as the substances that are used in the apparatus. The "settings", "readings" or "timings" of all the interacting components of the apparatus as well as their mode of interacting, and the inputs and outputs of the operation would be given in terms of energy and materials. This approach suggests how the *Flook* invention could be protected equally with the *Diehr* invention.

Filing and novelty

When an applicant can supply the proof to the Office that an initial act of commercialization of the technical innovation object has taken place, he can apply for a

---

[118] See this study, 6.3 [5] Subject-matter of the Utility Models. There is no preliminary examination to establish novelty and inventiveness in utility model system.
[119] Kingston, *Direct protection*, p. 263.
[120] Ibid., pp. 40-41.

provisional grant of protection.[121] This will be granted immediately if he can supply a declaration by a competent authority that the subject-matter is indeed novel, in the sense of "not being already commercially available".[122]

The Office will then publish the specification so that any interested third party may oppose the provisional grant. Since grant is irrevocable, unless it has been obtained through fraud, third parties are expected to submit the necessary information. The innovation patent office carries out its own independent examination.

In the classical patent system, a single document on its own can defeat a claim to novelty. In the Kronz system, such a document carries no influence at all since protection is not being given for a technical teaching, but for embodiments of teaching. A document will carry influence in the examination to the extent that it provides evidence of prior reduction to practice of the concept or teaching together with its actual use in public.[123] "New" does not refer to the teaching, but to the "act". This "act" (object, product or process) must be new within the jurisdiction. In the innovation patent system, "novelty" is based exclusively on "domestic public prior use" of a product or process available commercially.[124] The innovation patent is granted to the first innovator (doer) not to the first inventor (thinker).[125]

Claims

To define the scopes of the grant, Kronz uses "copy" or "option" claims.[126] "Copy" claim is single claim covering the innovation object in its precise concrete details,

---

[121] Ibid., p. 47.
[122] Kingston, *Innovation, Creativity and Law*, 1990, Kluwer Academic Publishers (hereinafter, Kingston, *Innovation*), p. 168.
[123] Kingston, *Direct protection* p. 48.
[124] Ibid., p. 269. Kronz admits that proving "prior public use" is more difficult than proving prior publication of the idea in the literature.
[125] Ibid., p. 260.
[126] Ibid., p. 49.

successively itemizing its elements, features and components, in a similar manner to "Jepson-type" claim.[127] The protection defined by a "copy" claim extends to technical equivalents, as in classical patents. Option claims cover alternative variants of the actual innovation object which has been the subject of the "first commercial act". The content of an option claim as such cannot be cited against the novelty of another innovation patent application. It becomes effective only after it has been embodied. Option claims will have weaker legal force than the copy claim. Licences, if requested, cannot be refused.

Infringement[128]

Since innovation protection is granted for the combination of the innovation object with its commercialization, any act of infringement of the protection must contain both of these elements. Mere manufacture of all the parts which would constitute an object which belongs to innovation patent does not constitute full infringement. And the act of simple selling the innovation object would not. These are an indirect infringement. Only when both types of infringement are combined, there can be an act of full infringement.

If the technical preparation has been done abroad, importing the product into a country which grants innovation patents constitutes the commercial step which completes the innovation act.[129] On the contrary, no infringement takes place simply by manufacturing for export to a country which does not grant innovation patents. In this

---

[127] Jepson-type claims are used to claim inventions that consist of improvements over existing articles, processes, or compositions of matter. After the description of a preamble that broadly describes all the conventional or known elements of the combination claimed, a description of the novel and non-obvious elements that constitute the new and improved portion of the claimed combination follows. See Jeffrey G Sheldon, *How to Write a Patent Application*, Practising Law Institute, November 1996.

[128] The definition of infringement in the patent statute encompasses making, using, selling, offering to sell, or importing a patented process or product. 35 U.S.C. 271(a).

[129] Kingston, *Direct protection*, p. 56. This point is somewhat questionable, when considering that simple selling the innovation object does not constitute infringement. In this case, first step (manufacturing) has been accomplished abroad.

case, there is only "latent infringement" in the country with innovation protection, since second (commercial) step is accomplished abroad.

Kronz thinks that an innovation patent is weaker than a classical patent in that it has only a reduced scope of protection. It protects against copying, including copying by the use of "technical equivalents".[130] An innovation patent also protects "optional" concepts disclosed by the patentee before the grant of the patent. Option claims do not deter their subject-matter to be used by third parties, since the claims have the obligation to grant a licence. The main purpose of option claims is to deal with the problem of infringement in cases where the supposedly infringing embodiment of the object of innovation is not a "technical" equivalent, but a "conceptual" one. In the innovation patent, the question of infringement can only arise when the subject-matter of the option claims is used without any licence. However, since granting a licence is obligatory, there should be very little litigation.

Kronz regards his system as being capable of supplementing or replacing the classical patent system. He thinks replacement as bringing the patent system back to its original value. Kronz accepts that if the patent is actually exploited, the innovation protection is not necessary.

[3] Kingston's Innovation Warrant

Like Kronz, Kingston as well has made the proposals for direct protection of innovation. Direct protection of innovation by warrants is achieved by making the subject-matter an investment which turns an idea into concrete reality. Warrants protect the investment which is concerned with getting new things done, where new information is generated.

---

[130] Ibid., p. 261.

In fact, anything new can be protected, as long as it can be the subject of investment, which means anything that can be bought and sold.

Subject-matter and novelty criterion

In the warrant system, the subject-matter of warrant protection is not an idea, but investment to turn an idea into concrete reality. New goods or services can be protected by the system, as long as it can be the subject of investment. If the subject-matter of the warrant application is not available for purchase now in the ordinary course of trade, an investment to make it available is entitled to the protection of a warrant.[131] Anything that can be bought and sold comes within the scope of the system, which extends far beyond technology. This means that computer software and methods of doing business can be protected. The criteria would be newness and the purchasability of the things for money. An important feature of Kingston's scheme is his emphasis on the national market. A product to be protected should be available in ordinary course of trade for the first time, and that means that it should be available through investment for production in the national market. (Imports would not satisfy the condition for the innovation warrant, while they would for Kronz.)[132]

Since the warrant system eliminates the argument of the "inventive step", it fits well with incremental innovations, such as computer software. For example, if a product with a particular new feature is not available in the ordinary course of trade and a product of a general type is available, an investment to bring the product with the particular new feature on the market is entitled to a warrant.

Novelty depends upon the answer to the question: "Is the subject-matter of the

---

[131] Ibid., p. 63.
[132] Ibid., pp. 203-204.

warrant application available for purchase now in the ordinary course of trade?" The criterion is applied only to a national market or a group of countries. That is, being "available in the ordinary course of trade" refers only to the national market. It enables the innovation office to ignore all evidences which do not come from the jurisdiction. A group of countries (e.g. A Community of countries), however, may move towards integrating their arrangements for granting this type of protection. They may extend their definition of novelty to "not available in the ordinary course of trade within the Community.

Features

The main features of the warrant system are public enforcement, incontestability and risk consideration. The enforcement of the warrant is rendered by the innovation office. This makes the quality of the monopoly completely independent of the warrant holder. There is particular logic to this approach. Since a warrant is granted by the state in order to encourage innovative investment, an attempt to infringe a warrant is regarded not just an act of damaging to a warrant-holder, but an attack on the economic policy of the state. The innovation office itself can prosecute infringers on behalf of the warrant-holder.[133] Kingston asserts that, considering the importance of a firm's investment at high risk in the generation of new information embodied in its product, the firm's asset arising from its efforts at innovation should be protected by the state, as in other types of property, e.g. money or buildings.

The warrant grant is incontestable unless it had been obtained by fraud, as in the Kronz system. Therefore, once a grant is made after opposition proceedings investment can be based upon it with complete confidence. Incontestability combined with freedom

---

[133] Ibid., p. 66.

from litigation would make a warrant attractive one for investment opportunities.

The term for the monopoly is determined by the consideration of the risk undertaken in an innovatory investment.[134] The length of the monopoly period is considered to match the reward to the risk in an investment. Kingston thinks that the more perfect the protection is, the shorter warrant terms can be. Kingston assigns probabilities of success of 0.5, 0.3 and 0.2 to incremental innovation, to technology-transfer-type innovation, and to radical innovation, respectively.[135] He also assigns similar probabilities to three levels of firm-related risk, i.e. low, medium and high. If the length of the term is to be the inverse of risk, the number of years will be the reciprocal of these probabilities. According to his illustration, a large firm carrying through an incremental innovation would have a monopoly of 4 years. A medium firm and a small firm carrying through an incremental innovation would have a monopoly of 6.7 years and 10 years, respectively. The innovation office should constantly carry out empirical investigations to improve statistical assessment of risk in its various categories. As the innovation office accumulates data and experience, its categories and warrant terms will become more and more effective in matching the monopoly term to the risk undertaken. Kingston uses categories of risk as the means of establishing the term of a warrant in order to eliminate administrative discretion of the innovation office.

Procedure[136]

A firm which wants to acquire a innovation warrant submits a proposal for investment to bring something new on the market, to the innovation office. The innovation may be a new product, or a product already on the market but with some new features. The

---

[134] Ibid., p. 62.
[135] Ibid., p. 68.
[136] Ibid., p. 76.

application should specify the amount of investment estimated to be required to carry through the innovation.

The office carries out an initial screening process to eliminate applications which are already on the market. There can be a right of appeal to the courts for the office's rejection. When the office's screening shows that the proposal is *prima facie* new to the market, its technical details is published, and a period is allowed for third party objections. Since a warrant is incontestable unless obtained by fraud, monitoring of such publication is very important to all firms.

After the opposition proceedings, if there is no ground for rejection, the office calculates both project-related and firm-related risks, and offers an option on a warrant for the appropriate term to the applicant.[137] When the innovation office offers a warrant to the applicant, some period of time (option period) is allowed for the applicant to make his detailed plan, arrange financing, and reach a final decision as to whether or not to make the necessary investment. The period may be related to the length of term of the offered warrant. It is expected that decisions as to radical advances and decisions involving more resources of a firm will take longer than incremental innovations.

The continuance of a warrant in force is conditional on making an investment to carry through the innovation, and this is time-bound. The investment should be completed within certain portion of the term, if the warrant is not to be nullified.[138] Stricter conditions would be given to the investment in incremental innovation and relaxed ones for the radical innovation.

---

[137] Ibid., p. 77.
[138] Ibid., p. 78.

Infringement

Kingston explains that "infringement" means "attempting to diminish the value of an innovation warrant, other than by innovation.[139] To ensure the effectiveness of warrant-holder's monopoly, Kingston suggests that it is necessary to develop a new doctrine of "commercial equivalence". This doctrine requires looking beyond the doctrine of technical equivalence which applies in the patent system. This is because there are many ways where a competitor might diminish a warrant-holder's possibility of recouping investment, other than by producing a product which is technically equivalent. The criterion will depend on the answer to the question: "If the alleged infringer's product had not been available, would the warrant-holder have made a particular sale?" Such protection is wider than that of a patent, but it is limited in two main ways. Firstly, any product which was on the market at the time of the application for a warrant, cannot be affected. Secondly, any product that is the subject of an innovation warrant, cannot be held to infringe any other warrant.

To protect warrant-holders, the innovation office should take action against competitors producing "commercially equivalent" products.[140] A different innovation which itself obtained a warrant will not be attacked under this procedure. However, a new warrant-holder may have to pay a royalty to an earlier warrant-holder, if his innovation is based upon the earlier one.

[4] Implications of Protecting Innovation Directly

The two proposals, the innovation patent by Kronz and the innovation warrant by

---

[139] Ibid., p. 74.
[140] Ibid., p. 205.

Kingston, were developed independently of each other.[141] They, however, have the following common features:

- The subject-matter of protection should be innovation, not invention.
- Any economic object, including technology, can be protected.
- The criterion of novelty should be actual commercial availability.
- The term of grant should be variable.
- Grants should be incontestable unless obtained by fraud.
- Terms of grant may differ between regions of a country.
- Examination is relied heavily on third party.
- The system can be administered by an independent authority.

Their main differences are the following:

- The innovation patent requires that innovation object should exist before protection is given. This means that the associated investment should be made first. The innovation warrant offers protection before investment is made. Actual investment is the condition of keeping the monopoly in force.
- The innovation patent system tries to match the length of monopoly to the individual innovative capacity of the patentee. However, the innovation warrant system seeks to eliminate official discretion and makes a set of terms which may not correspond to the innovative capacity so exactly in each cases.
- The innovation patent office prescribes licensing terms that consider the innovative capacity of the licensee. The innovation warrants has no similar provisions.
- While the patentee of an innovation patent should protect his own right, in the

---

[141] Kingston argues that they were developed completely independently of each other. See, ibid., p. 87.

warrant system the state should protect the warrants it makes.

- The scope of protection is determined by "technical equivalents" in Kronz system, and "commercial equivalents" in Kingston system, respectively. The doctrine of "commercial equivalence" extends beyond "technical equivalence".

Firm size

While the shift from direct to indirect protection of innovation by the patent system has given much benefit to large firms, it has not provided secure protection to SMEs.[142] This shift has made many SMEs avoid investment in innovation. This has serious consequence because SMEs are often more suited to innovation than large firms.[143] Both proposals consider firm size from the outset and the protection becomes equal for large firms and small ones. Particularly, the elimination of the warrant-holder's burden of protecting his right makes the differences in their size and their capacity to pursue litigation irrelevant. Due to the monopoly conferred by its innovation patent or warrant, SMEs are now the equal of the large firms as far as a particular innovation is concerned.

Incremental innovation

Incremental innovation is most likely to be achieved in the small firm.[144] Incremental innovations are mostly the improvements to products or processes which are individually small, but cumulatively of supreme economic importance.[145] They may be obvious to one skilled in the art, and therefore unpatentable under the present patent system.

---

[142] Ibid., pp. 99-100.
[143] SMEs account for more than 99% of all European firms, 66% of all jobs and 65% of turnover in the EC. See this study, 6.3 [4] Implications of the Utility Model Regime.
[144] Kingston, *Direct protection*, p. 107.
[145] Kingston, *The Political Economy of Innovation*, 1984 Martinus Nijhoff Publishers, p. 211.

Kingston asserts that direct protection for many incremental innovations can be achieved by the two proposals. The classical patent requires that an invention should have an inventive step and it should not be obvious to one skilled in the art. It is indeed true that the majority of patents are for small "improvements".[146] The countless small changes in a product are those which are underlying in the preceding technology, and grow naturally out of it. This "natural" or "evolutionary" growth makes them obvious and unpatentable. This means that the patent system does not protect investment in incremental innovation. By giving up the "inventive step" criterion, he maintains, incremental innovations can be effectively protected. Kronz as well argues that the abandonment of the inventive step criterion is substantially what the innovation patent system proposes.[147] Considering the fact that "modifications" of something which already exists can obtain an innovation patent if it is neither a "technical equivalent" nor an object of an option claim of the initial innovation patent, innovation patents will be much the same as *utility model* in number and level.[148]

In the case of a second innovator who has brought an incremental change to a protected product, protection will be granted to the second innovator for the incremental change, but the grant will be endorsed with requirement that the innovation cannot be put into practical effect without infringing the first innovator's right, since it will use some of the information generated by the first innovation.[149] The use of the incremental improvement will depend on agreement between the two parties. The first innovator will naturally want the incremental improvement by a second innovator to be

---

[146] Kingston, *Innovation,* p. 173.
[147] Kingston, *Direct protection*, p. 267.
[148] Ibid., pp. 270-271. The lower novelty and inventive step requirement of utility models provide protection for small technological advances. See this study, 6.3 [4] Implications of the Utility Model Regimes.
[149] Kingston, *Direct protection*, p. 313.

incorporated in his product as soon as possible. The second innovator may come to an arrangement with the first innovator to permit this to benefit from his innovation immediately, and he may use the information of the first innovation with paying royalty to the first innovator. The second innovator may have another choice. He can wait until the first innovator's monopoly is expired.[150] Then he will be free to use the first innovation with his own improvement incorporated in it.

Know-how

Direct protection may also stimulate technology transfer by providing protection of know-how.[151] The existing patent system is supposed to do this, but in reality fails to do so. For examples, studies of licence agreements show that important thing is know-how, not what is disclosed in the patent specification. A reason why "know-how" is lacking in the patent specification is that the inventive step criterion does not permit the protection of a craftsman's practical knowledge.[152] If know-how receives no protection under the present patent system, no applicant will want to disclose it. However, because both innovation patent and warrant give protection to know-how, there is no reason why applicants will be unwilling to make these disclosures in exchange for protection.

Competition policy

The monopoly granted by an innovation patent or warrant will be a sound basis for establishing a new firm, or expanding an existing one, to carry through an innovation. This will reduce the harmful effects of the concentration of market power. The innovation patent or warrant will provide insurmountable barriers to entry. Even when

---

[150] This point is also found in the Compensatory Liability Regimes by Reichman.
[151] Kingston, *Innovation*, p. 175.
[152] Kingston, *Direct protection*, p. 267.

the term has expired, large firms may still find that their entry is made difficult by the market power which the smaller firm has established during its monopoly period. Thus, direct protection forces larges firms to be more innovative than they are now.

Infringement and litigation

To determine whether or not an object belongs to the scope of the protection of an innovation patent or an innovation warrant, it is necessary to decide whether the object is a "technical equivalence" or a "commercial equivalence". The scope of "commercial equivalence" is much wider than that of "technical equivalence" and much difficult to determine. It includes the consideration of the time and the market which the product belongs to. Even though an object is not a technical equivalent, it can be a commercial equivalent. In this respect, there could be many disputes. Moreover, "commercial equivalence" may change depending on time. As time goes by, an object which constituted infringement before may become other than a "commercial equivalent", according to changing market, and vice versa.

Claims in innovation patent are unlikely to cause litigation.[153] Infringement of a copy claim will be a very unusual event, because such a claim covers the actual embodiment marketed by the patentee, with protection extending to technical equivalents only. A competitor can follow by "innovating around" a copy claim by changing the product in the ways that are not technically equivalent. Alternatively, competitor can seek a licence under one of the option claims. The option claims cover alternatives which have been considered and tested by the innovator. A licence to use the alternatives covered by option claims must be granted, if requested. (In this respect, the two proposals are similar to the Compensatory Liability Regime proposed by

---

[153] Ibid., p. 325.

Reichman.) Thus, infringement may be regarded as an unimportant feature for the innovation patent regime. On the contrary, infringement is very important for the innovation warrant.

Moreover, the innovation warrant proposals lift the burden of enforcing the monopoly grant from the warrant-holders. The typical problem of litigation is that it takes too much economic resources. If the warrant-holder cannot enforce without going to courts, and if he has no resources to do this, the protection is actually worthless. This is why the warrant proposals make the enforcement the responsibility of the granting authority. By contrast, in the innovation patent, enforcement remains the patentee's own responsibility and infringement will be pursued through the courts in the same way as in the case of classical patents.

Changes in term arrangements

As a result of experts' comments, Kingston and Kronz improved their proposals in several respects. Particularly, the individual "variable term" arrangements of both innovation patent and innovation warrant have been replaced by three fixed "project-related" terms.

Comments on subject-matter of the direct protection

It is much better to give protection to an existing object or process that is new, than to the idea which can often be embodied in various ways. This is because ideas can be embodied in diverse economic objects, and they can be easily litigated.[154]

To give protection beyond technological products or processes may include the field of fashion, designs, organizations, sales methods, services (computer programs,

[154] Ibid., p. 225.

information systems, management practices, etc) and the like. Thus, the innovation protection covers computer programs and methods of doing business.

Considering the importance of incremental innovations, especially in the fields of computer software, abandonment of the non-obviousness criterion in the innovation protection may be helpful for the protection of innovations of computer software.[155]

6.5 Self-Help Systems

'Self-help' refers to an expanding set of technologies and systems designed to protect content from unauthorized copying and to facilitate e-commerce involving content.[156] Kenneth W. Dam, the author of 'Self-help in the Digital Jungle' uses 'content' broadly to include "text, data, images, audio, video, and all of the other media that patrons of the Web are familiar with". He uses the concept of a 'content' in the broadest possible sense to include 'all forms of information and without distinction as to whether or not the information is legally protected against access by unintended recipients through IPRs'.[157]

He asserts that self-help systems will not only reduce the incidence of copyright violations but be one of the crucial success factors in e-commerce. He maintains that because the systems can protect uncopyrightable or uncopyrighted materials as well as copyrighted materials, they should not be viewed as conflicting with the IP law of copyright.

Self-help systems enable a content provider to transmit content to a potential reader by posting it on a website, e-mailing it, etc. while preventing anyone from

---

155 In the innovation patent the criterion of inventive step is no longer applied.
156 Dreyfuss, *Expanding*, pp. 103-104.
157 Ibid., p. 107.

accessing it without permission. The systems can facilitate implementation of many of the ideas underlying pro-competitive and fair use ideas embedded in IP law.[158] Self-help systems can harness the characteristics of digital copies that they are normally identical with one another. The technology of self-help systems lowers transactions costs and thereby reduces undesirable social behaviour (such as free riding appropriation of content created by others). As transaction costs go down (and convenience goes up), it is easier for people to do the right thing (that is, paying or obtaining permission).

According to the research sponsored by the UK Economic and Social Research Council, under the £1.2 million program *Intellectual Property Initiative*, SMEs relied generally on copyright for their software.[159] 100% of the firms interviewed resorted to copyrights as their main mode of protection because it is cheap and automatic. They also employed several informal methods of protection, particularly technical systems, such as encryption,[160] dongles,[161] steganographic techniques,[162] key diskettes,[163]

---

[158] Ibid., p. 110.

[159] Adams and Tang, p. 18. Their general reliance on copyright appears to be based on the characteristic of copyright, i.e. automatic protection unnecessary for formal registration.

[160] http://searchsecurity.techtarget.com/sDefinition/0,,sid14_gci212062,00.html accessed 16 October 2001. Encryption is the conversion of data into a form, called a ciphertext, which cannot be easily understood by unauthorized people. Decryption is the process of converting encrypted data back into its original form, so it can be understood.

[161] http://www.computerlanguage.com/sitemain/content.html accessed 16 October 2001. Dongle is the same as hardware key. Hardware key: Also called a "dongle," it is a copy protection device supplied with software that plugs into a port (parallel, serial, USB, etc.) on a PC. The software sends a code to that port, and the key responds by reading out its serial number, which verifies its presence to the program. The key hinders software duplication, because each copy of the program is tied to a unique number, which is difficult to obtain, and the key has to be programmed with that number.

[162] Steganography: Hiding a message within an image, audio or video file. Used as an alternate to encryption, it takes advantage of unused bits within the file structure or bits that are mostly undetectable if altered. A steganographic message rides secretly to its destination, unlike encrypted messages, which although undecipherable without the decryption key, can be identified as encrypted. For a white paper on the subject written by Neil F. Johnson of George Mason University, visit www.jjtc.com/Steganography.

[163] http://www.computerlanguage.com/sitemain/content.html accessed 16 October 2001. Key: In cryptography, a numeric code that is combined in some manner with the text to encrypt it for security purposes.

firewalls[164] and passwords.[165] More than half of the respondents used these systems. The research shows that, while only a minority of SMEs patent their software creation, the majority of them regard copyright, technical systems of protection, and licensing as the most common methods of protection.[166] The survey data on how SMEs rank the importance of methods of IP protection reveal that 27% of SMEs ranked licensing as the most important means of protection; 24% of them regarded technical systems of protections as the most important; and 21%, copyright. It is interesting to note that 52% of those interviewed considered market niche and technical systems of protection as important methods of protection.[167]

Contrary to the wide use of technical methods (self-help systems against circumvention), several respondents, even though they were using the technical methods, acknowledged that such systems are generally user unfriendly, and complicated. Those who did not employ these methods explained the reason that the lack of an industry standard made them cautious about employing them.[168] Other respondents regarded the push for technical protection as a conspiracy by large companies to protect their territories from more innovative and imaginative smaller companies. Similarly, the Legal Advisory Board (of the European Commission) stated that the widespread use of technical protection devices might result in the *de facto* creation of new information monopolies. These views appear to emphasize the necessity to protect fair use rights.

---

[164] Firewall: A method for keeping a network secure from intruders. It can be a single router that filters out unwanted packets or may comprise a combination of routers and servers each performing some type of firewall processing. Firewalls are widely used to give users secure access to the Internet as well as to separate a company's public Web server from its internal network. Firewalls are also used to keep internal network segments secure; for example, the accounting network might be vulnerable to snooping from within the enterprise.

[165] Adams and Tang, pp. vi-vii, 19.

[166] Ibid., pp. viii, 58.

[167] Ibid., p. 19.

[168] Ibid., p. 19.

Encryption

Encryption is the basic technology of self-help systems. The encrypted content is placed within a digital envelope so that the content provider can indicate in unencrypted text on the envelope what a reader has to do decrypt the content.[169]

Digital watermarks

A digital watermark[170] can be placed on an image so that any copies can be identified as being originated from the content provider or as being copied from an image transmitted to a specified party.[171] This discourages sending the copy on to a third party who might make copies unauthorized by the content provider. Digital watermark technology can be combined with a search program that wonders the net and looks for the provider's watermark, finding out unauthorized use of the content.

Contrary to the common misunderstandings about self-help systems, watermarks are not just for content providers, but they can benefit users. For example, a user of a program can determine the source of a watermarked photo and he can communicate directly with the original photo owner.

Invisible messages

Self-help systems can attach invisible messages to content, which make it impossible to copy the content, or allow only a single copy, or send a message back to the content

---

[169] Dreyfuss, *Expanding*, pp. 107-108.

[170] http://lookup.computerlanguage.com/host_app/search accessed 16 October 2001. Digital watermark: A pattern of bits embedded into a file used to identify the source of illegal copies. For example, if a digital watermark is placed into a master copy of an audio CD or a DVD movie, then all copies of that disc are uniquely identified. If a licensee were to manufacture and distribute them in areas outside of its authorized territory, the watermark provides a trace.

[171] Dreyfuss, *Expanding*, p. 108.

provider indicating how many copies are being made.[172] Locking mechanisms can be classified as a kind of invisible messages. Content can be locked so that it has to be unlocked by each recipient. If the content provider transmits content to an original recipient who unlocks it by payment and then forwards it to a friend, the friend will receive a locked copy and cannot unlock it without paying.

The foregoing are just a few variations on the concept of a self-help system. Almost any conceivable combination or variation of the ideas discussed above is possible.

Vulnerability of self-help systems

Self-help systems are vulnerable to attack, like any electronic online system. Computer programs can be written to detect and strip off invisible messages. It is anticipated that development of software technologies makes it possible to detect digital watermarks and to wash them out.

Fair use and self-help

The Digital Millennium Copyright Act enacted in October 1998 takes the first step in addressing the relationship between fair use and self-help systems. It applies only to copyright and leaves open the question of noncopyrightable content. Recognizing the vulnerability of self-help systems, the Act prohibits circumvention of any 'technological measure that effectively controls access' to a copyrighted work as well as the manufacture, importation, or offer to the public of any technology primarily produced

---

[172] Ibid., p. 109.

for the purpose of such circumvention.[173] But since such measures against circumvention may affect the exercise of fair use rights, the statute establishes a system for determining whether users of particular classes of works are adversely affected due to such prohibition in their ability to make noninfringing uses of those particular classes of works.[174] Users of such classes of works are not subject to the circumvention prohibition. Six categories which include criticism, comment, news reporting, teaching, scholarship and research are the kinds of potential fair use.[175]

Moral rights and deterrence

Self-help systems can serve the purposes of moral rights. Firstly, they can assure attribution to the author, artist, or composer. Secondly, they can ensure the integrity of documents, images and music.

Self-help systems help protect against liability. Problems involving alteration of evidence in litigation can be avoided by time stamps to documents through invisible messages that can only be removed by a determined attacker.

Self-help systems can also protect artists who do not use self-help systems. Pirates, if they know that watermarks are being used to trace piracy, would choose those artistic works without a watermark and avoid those with a watermark. However, since watermarks are invisible, piracy of all artistic works will be deterred, not just those with a watermark.

---

173 17 U.S.C. 1201(a)(1)(A). Dreyfuss, *Expanding*, p. 112.
174 17 U.S.C. 1201(a)(1)(B).
175 17 U.S.C. 107. Dreyfuss, *Expanding*, p. 112.

Self-help and social norms

Technology can promote ethics and the public good by reducing transactions costs.[176] The technology of self-help systems lowers transactions costs especially when combined with digital cash through increasing the convenience of payment. This reduces undesirable social behaviour such as free riding appropriation of content created by others. As costs go down, it is easier for people to do 'what is right' (that is, paying or obtaining permission for copying content created by others). As more people do this, others are more likely to follow suit and thus establish a custom of what is expected and acceptable behaviour.

Subject-matter of the self-help systems

'Self-help' refers to technologies and systems designed to protect *content* from unauthorized copying.[177] Dam, the author of 'Self-help in the Digital Jungle' uses 'content' to include *text, data, images, audio, video, and all of the other media that patrons of the Web are familiar with.* He also uses the concept of a 'content' in the broadest sense to include *all forms of information and without distinction as to whether or not the information is legally protected against access by unintended recipients through IPRs.*

---

[176] Dreyfuss, *Expanding*, p. 119. In an effort to reduce transaction costs, patent pools have been developed.
[177] Ibid., pp. 103-107.

CHAPTER 7

Evaluation, Conclusion and Suggestions

7.1 Evaluation

In spite of many debates on the desirability of extending the patentability of software creations, software patenting is increasing especially with the rapid growth of e-commerce.

According to the economic review of the desirability of patenting software and business methods, the existing proprietary regimes (i.e. patent, copyright law and trade secrecy) do not provide appropriate protection for software innovation. Patents can protect the idea or concept in computer programs, which may have great value. As the importance of doing business on the Internet grows, the need to protect the software by patents grows. The novelty and inventive step requirement of patent law, however, prevents software innovations, which are essentially incremental and cumulative, from patent protection. On the other hand, if patents are granted to the incremental innovations by rewarding inventors with strong exclusive property rights, the patents will impede the process of follow-on innovations. Copyright protection for software is convenient because it is automatic. However, it does not protect program behaviour and does not protect the idea or concept in software. It protects only the specific form where the idea is expressed. Trade secrecy cannot protect much of the know-how used in software design, since such know-how is largely evident in distributed products and trade secrecy cannot protect what is not secret.

These problems of traditional protection for software lead us to alternative

proposals: (1) A market-oriented legal regime, (2) Compensatory liability regimes, (3) Utility models, (4) Direct protection of innovation, and (5) Self-help systems. In order to find the most appropriate form of protection for software, it is necessary to evaluate these alternatives in the light of the economic perspective and the development of software. It is also useful to consider the design principles and goals discussed in the market-oriented regime (e.g. to build on existing legal foundations, to focus on the most serious problems, to be responsive to the characteristics of software, to encourage innovation and so on).[1] The most appropriate protection would not only solve the most critical problems (i.e. discouraging follow-on innovations or causing market failure in the existing legal regimes), but also reflect the characteristics of the development of software and satisfy as many design principles as possible.

[1] A Market-Oriented Legal Regime

A market-oriented legal regime describes a number of design principles and goals which can be a basis of a new form of legal protection for software.[2] The market-oriented regime provides a two-part solution: (1) a protection scheme organized around the source of value in the software, i.e. program behaviour, and (2) a protection scheme based on principles of market economics. There are a number of possible legal mechanisms for implementing a market-oriented approach. The authors of *A Manifesto* think that the approach that appears to match best with the design principles is the one that would provide software developers with both a market-preserving period of protection against cloning and a period of time within which to register their program

---

[1] See this study, 6.1 [3] Goals and Principles for a Market-Oriented Approach.
[2] See *A Manifesto*.

design innovations. Registration would provide compensation for the use of the innovation by a second comer for a period of time after the expiration of the anti-cloning blocking period.

The market-oriented regime of which subject-matter is 'selection and arrangement of useful components', and 'compilations of behavioural components', well addresses the characteristics of modern software development, i.e. componentization and re-use. The Automatic Anti-Cloning Protection Followed by an Automatic Royalty-Bearing Licence system (Anti-Cloning Automatic Licence system) in the market-oriented regime provides the innovator with some compensation from others who use his innovation in the second phase, i.e. the automatic licence period, regardless of commercial success. The Anti-Cloning Automatic Licence system appears to be convenient and appropriate for the protection of software innovations in that the speed of software innovation is fast and the lifecycle of software products is short. However, it is difficult to determine the appropriate period of automatic anti-cloning protection, as well as the period of and licence fee in the automatic royalty-bearing licence. The period must be proportionate to the lead-time necessary to give individual innovators the opportunity to develop a market niche. Moreover, because there is no registration system, it is difficult to identify what the subject-matter is protected, and when the automatic licence period commences.

The Automatic Protection Complemented by Registration of Innovative Elements system (Automatic Protection Registration system) requires registration that would give an extended period of exclusivity or an automatic royalty-bearing licence.[3] Registration might be available on standard terms after expiration of the unregistered protection right. The registration system would help establishing a documented prior art. The automatic

[3] Automatic royalty-bearing licence is similar to the concept of Reichman's Liability Regime.

royalty-bearing licence would remove the transaction costs of licensing. However, the Automatic Protection Registration system has problems in that SMEs do not favour any formal registration system because they view formal legal rights, particularly those requiring registration, as expensive, time-consuming, complex and of limited value.[4]

Critique

The market-oriented regime falls short in identifying the concrete method to protect the conceptual metaphors to organize behaviour and bring about a synthetic reality, i.e. *virtuality*, even though the authors of *A Manifesto* regard them valuable. The main problem with the regime is that instead of providing a detailed implementation scheme, it has only a basic framework for constructing a new form of legal protection for software innovations.[5]

[2] Compensatory Liability Regimes

Reichman's compensatory liability regime correctly addresses the critical nature of software innovations, i.e. cumulative and incremental improvements based on componentization and re-use.[6] This is because the liability regime tries to protect cumulative and sequential working out of common technical trajectories as well as sub-patentable (or small scale) innovations that do not rise to the level of novelty and non-obviousness to be patentable inventions.

Within a designated period of artificial lead-time, firms are allowed to borrow one another's sub-patentable innovations, only when they contribute to the costs of

---

[4] http://info.sm.umist.ac.uk/esrcip/Projects/L5253004/final_report.htm accessed 8 October 2001.
[5] *A Manifesto*, Introduction.
[6] Dreyfuss, *Expanding*, p. 29. .

development. Second comers do not have to negotiate permissions. This reduces the transaction costs. The automatic licence in the compensatory liability regime may minimize the unjustifiable tendency of exclusive property rights to allocate ownership of follow-on applications either to the first comer or to second comers. This approach would provide sub-patentable innovators with enough lead-time to recoup their investments and make sufficient profits to enable further investments.[7] Thus, this regime solves the problem of appropriability in order to encourage investment without necessarily entitling the first comer to all the returns from follow-on innovations.[8] At the same time, this alternative would neither retard scientific research, nor hinder follow-on innovations, nor create legal barriers to entry. A properly crafted liability rule would offer those who innovate a way to alleviate market failure.[9]

Critique

One of the problems of the liability regime is that it does not provide a detailed implementation scheme, e.g. an exact definition of sub-patentable innovations, constituents of infringement, and registration procedures. Reichman does not provide concretely the definition of what makes sub-patentable innovations. Definitions such as 'small scale (or sub-patentable) innovations that do not rise to the level of novel and non-obvious inventions or original and creative works of authorship' and 'cumulative and sequential working out of shared or common technical trajectories' are vague. There have been many debates about the issue of how an invention can be characterized as novel, and it is much more difficult to determine the level of non-obviousness,

---

[7] J.H. Reichman and Pamela Samuelson, *Intellectual Property Rights in Data?* Vanderbilt Law Review, January 1997, 50 Vand. L. Rev.51. http://eon.law.harvard.edu/h2o/property/alternatives/reichman.html accessed 21 July 2001.
[8] Dreyfuss, *Expanding*, p. 29.
[9] Ibid., p. 51.

especially in software industry. It is not specified what the lowest level of sub-patentable innovations which qualify as the subject-matter of the liability regime is. This needs to be discussed because not all kinds of selections, adding, rearrangements, modifications and adjustments should be protected. There should be minimum requirements to be regarded as sub-patentable.[10] These aspects are closely related to the issue of infringement. For examples, simple change of components which are equivalents, or simple modifications of known elements without any resulting effects, should not be regarded as the subject-matter of liability protection. The problems with registration have been discussed above.

For the compensatory liability regime to be employed as an actual protection system, these problems need to be addressed.

[3] Utility Models

The main features of the utility model, compared with a patent, are a lower level of inventiveness than that required for a patent, the absence of prior examination, and a short protection period. These features appear to reflect the characteristics of software innovations.

Critique

The main problem with the utility model is the fact that its subject-matter is mainly devices relating to the shape, or construction of articles, or a combination of such things.

---

[10] When an invention could easily have been made prior to the filing of the patent application by a person having ordinary skill in the art to which the invention pertains, a patent shall not be granted for such an invention. Korean Industrial Property Office, 1998, *Industrial Property Law of the Republic of Korea*, Patent Law, Article 29 [Requirements for patents].

Methods of construction or articles not having a certain shape are excluded from utility model protection. Thus, protecting software by the utility model system appears to be inappropriate, since software has no shape and is an execution of complex logic. However, in the sense that programs are machines,[11] and that writing programs is an industrial compilation[12] of sub-components which is similar to the design of physical machines, software can be regarded as a device having *virtual* shape, and the system might be adapted accordingly.

Another important problem of the utility model, however, is that it is a proprietary right. The exclusive property regimes uniformly impose on the process of follow-on innovation unacceptably high social costs.[13] An exclusive property regime fails to solve the problem of follow-on applications of sub-patentable know-how to marketed products. Any system that protects sub-patentable applications of technical know-how by means of a property right will tend to reward individual innovators as if they had produced major innovations.[14] That is, by rewarding individuals with strong exclusive property rights for routine applications of the community's technical know-how, the system tends to make that shared know-how artificially scarce. As the tiny bundles of small-scale innovation covered by strong IPRs and contractual rights increase, the community's shared know-how is divided into ever smaller pieces which are withdrawn from the public domain. This process constitutes a tangled web of property rights and a barrier to entry as well as a disincentive to further small-scale innovation. The need to bargain around an exclusive property right complicates usual business transactions and adds new risks of infringement litigation to the inherent risks of predicting market

---

[11] See this study, 5.2 [2] Programs are Machines.
[12] See this study, 5.2 [4] Programs are Industrial Compilations.
[13] Dreyfuss, *Expanding*, p. 28.
[14] Ibid., pp. 37-38.

success. In sum, property-based rules impede follow-on developments, ignore the significant contributions of the public domain, and increase transaction costs.

In the computer software industry, the patent (and copyright) system is creating a *patent thicket*.[15] The vast number of patents currently being issued creates a real danger that a single product or process will infringe many patents.[16] Moreover, many patents cover products or processes already being widely used when the patents are issued, and they make it harder for the companies actually manufacturing products to invent around the patents. Furthermore, a patent holder can seek injunctive relief, i.e. can threaten to shut down the operations of the infringing company. There have been many concerns about a patent thicket being created by BMPs. In this state of affairs, the introduction of the utility model that is also a proprietary right would mean establishing *utility model thicket*[17] on the top of patent thicket. It would make matters worse.

As for the issue of the lower level of inventiveness in the utility model, it is almost the same as that of non-obviousness in sub-patentable innovations discussed above.

[4] Direct Protection of Innovation

Direct protection of innovation aims at protecting innovations rather than inventions. Innovation patents or warrants would be given to the initial act of commercializing computer programs.[18] Direct protection by an innovation patent is similar to copyright

---

[15] Adams Jaffe, *Innovation Policy and the Economy*. 'Patent thicket' means an overlapping set of patent rights requiring that those seeking to commercialize new technology obtain licences from multiple patentees. Cross-licensing and patent pools are two natural and effective methods used by market participants to cut through the patent thicket.

[16] Almost all authors of software will involuntarily infringe a software patent when they publish their software. See this study, 5.4 [2] Cons for Software Patents.

[17] The term '*utility model thicket*' is devised by the author in order to mean the *thicket* created by a great number of utility models.

[18] Direct protection of innovation by warrants is achieved by making the subject-matter an investment

which protects the work *itself*.[19]

The direct protection effectively protects incremental innovations, which become cumulatively more important in the modern software industry, by giving up the "inventive step" criterion.[20] Many incremental innovations such as *transposition, application, identification, formulation, selection, simplification, combination* and *aggregation* of software components can be protected by the innovation patent. Modifications of software components which already exist can obtain an innovation patent if it is neither a technical equivalent nor an object of an option claim of the initial innovation patent.[21] In other words, in the case of a second innovator who has brought an incremental change to a protected product, protection will be granted for the incremental change (with the requirement that the innovation cannot be put into practical effect without infringing the first innovator's right). The use of the incremental improvement will thus depend on agreement between the two parties. The relationship between the first comer and second comers is similar to that of the liability regime.[22] Thus, the direct protection system solves the problem of how to enable companies to appropriate the fruits of their investment in sequential innovation without impeding follow-on innovations.[23]

In the direct protection system, fear of litigation greatly diminishes compared with the classical patent system. A competitor can follow by innovating around a copy claim by changing the product in the ways that are not technically equivalent. Alternatively, he can seek a licence under one of the option claims. A licence to use the alternatives

---

which turns an idea into concrete reality.

[19] Kingston, *Direct protection*, p. 39. Copyright does not protect any idea or concept for the work which an author might have in mind.

[20] Ibid., p. 267.

[21] Ibid., p. 271.

[22] See this study, 6.2 Compensatory Liability Regimes.

[23] Dreyfuss, *Expanding*, p. 23.

covered by option claims must be granted. Moreover, in the warrant system, any product which was on the market at the time of the application for a warrant, cannot be affected and any product which is the subject of an innovation warrant, cannot be held to infringe any other warrant.

The elimination of the warrant-holders' burden to protect their rights makes the differences in their size and their capacity to pursue litigation irrelevant. Due to the monopoly conferred by their innovation patent or warrant, SMEs are the equal of the large firms as far as a particular innovation is concerned. This is very desirable because SMEs are great source of innovations,[24] and because this makes it easy to establish a new company and makes large companies more innovative.

Moreover, the innovation patent and the innovation warrant are incontestable unless obtained by fraud.[25] Incontestability and freedom from litigation (in the innovation warrant) can be a good base for investment.

Critique

One of the main problems of the Kronz system exists in the examination of the novelty criterion. A document carries weight in the examination to the extent that it provides evidence of prior reduction to practice of the concept, together with its actual use in public.[26] It would be very difficult for an examiner to find a document with the evidence of prior reduction to practice and actual use in public, since there is no established prior art of this kind. Establishing prior art with such evidence would be a difficult job. Without an established prior art, accurate examination of the innovation

---

[24] See this study, 6.3 [4] Implications of the Utility Model Regime.
[25] However, due to the incontestability, the problem of bad innovation patents exists when innovation patents were granted to already embodied innovations.
[26] Kingston, *Direct protection*, p. 48.

patent applications would be unimaginable. There is a similar problem in the innovation warrant system. In the innovation warrant, novelty depends on the purchasability of the subject-matter of a warrant application in the ordinary course of trade. The purchasability of software differs depending on time. Software that was purchasable may become otherwise as time goes by. Finding a prior art with the evidence of the purchasability and establishing a prior art with such evidence for the examination are also a difficult job, especially when combined with the typical problems with the prior art in the software industry.[27]

In determining whether or not an object belongs within the scope of the protection of an innovation warrant, it is necessary to decide whether the object is a "commercial equivalent." However, as time goes by, an object which constituted infringement before may become other than a "commercial equivalent", according to the changing market, and vice versa. Moreover, when it comes to computer programs, the scope of "commercial equivalence" becomes more broad and vague. It is difficult to determine whether a computer program belongs within the scope of "commercial equivalent" of another computer program to which a warrant is granted, because it includes consideration of the time and the changing market which the software product belongs to. Furthermore, considering the realities of the development of software, the criterion of "commercial equivalent" is so broad as to include almost all incremental improvements which have been made on the top of existing software products. In this respect, there would be many disputes.

The two proposals by Kronz and Kingston, however, are so well equipped with details of their implementation that they may be regarded as being capable of supplementing or replacing the classical patent system in this field. Moreover, they are

---

[27] See this study, Chapter 5. Economic Review of Protection of Computer Programs by Existing Regimes.

so similar to the classical patent system that existing patent offices could take over the job of the direct protection system without establishing an Innovation Office separately. The patent examiners would be able to deal with the innovation applications with some training. The public as well would not have much difficulty in applying for an innovation patent or warrant, because they are already accustomed to the classical patent system. These features enhance the feasibility of the two proposals.

[5] Self-help Systems

Self-help systems will reduce the incidence of copyright violations. They are one of the crucial success factors in e-commerce. The systems can facilitate implementation of many ideas underlying pro-competitive and fair use concepts embedded in IP law.[28] Self-help systems can harness the features of digital copies that are generally identical with one another. The technology of self-help systems lowers transactions costs and thereby reduces undesirable social behaviour such as free riding appropriation. As transaction costs go down, it is easier for people to do the right thing. Self-help systems are not just for software providers, but also for the users' benefit.

The majority of SMEs regard technical systems of protection as well as copyright and licensing as the most useful methods of protection.[29] The survey[30] on how SMEs rank the importance of methods of IP protection reveals that 24% of SMEs regarded technical systems of protection as the most important and 52% of those interviewed considered market niche and technical systems of protection as important methods of

---

[28] Dreyfuss, *Expanding*, p. 110.
[29] Adams and Tang, pp. viii and 58.
[30] The research sponsored by the UK Economic and Social Research Council, under the £1.2 million program *Intellectual Property Initiative*.

protection.[31] This data suggests that self-help systems are an appropriate protection form for software.

Critique

The main problem of Self-help systems is that they are vulnerable to attack. Development of software technologies would make it possible to detect digital watermarks and to wash them out. In spite of their advantages, depending on self-help systems completely may result in market failure, especially when software developers are SMEs which have a limited ability to use the technology of self-help systems, and when large companies, which have enough resources to make the self-help technology useless, are trying to free ride the computer programs developed by the SMEs. Thus, the self-help systems would need a supplementary form of protection.

7.2 Conclusion

In conclusion, while the market-oriented regime provides a basic framework for constructing a new form of legal protection for software, the regime does not have enough details for a model statute.[32] Reichman's compensatory liability regime solves the critical issue of the relationship between the first comer and second comers in sequential innovation, i.e. encouraging innovation without impeding follow-on innovations. The liability regime, however, does not provide detailed implementation proposals. Thus, both the market-oriented regime and the liability regime are not feasible in the near future. The utility model makes the patent thicket more complex by

---

[31] 27% of SMEs ranked licensing as the most important means of protection; and 21%, copyright. Adams and Tang, p. 19.

[32] *A Manifesto*, Introduction.

establishing *utility model thicket* which impedes follow-on innovations. Thus, the utility model system does not solve the most serious problem of the existing legal regimes. The direct protection of innovation not only solves the most serious problems, but also satisfies more goals and principles for a market-oriented approach than any other alternative. For example, the system effectively protects incremental innovations by solving the most critical problems of the classical patent system which discourages software innovation by impeding follow-on innovations. The system can be established on existing legal foundations. The system has many other advantages such as reduced fear of litigation, elimination of the warrant-holder's burden of protecting his right, incontestability, feasibility (due to a detailed scheme of implementation) and familiarity (of examiners and the public).

Consideration of these advantages leads us to choose the direct protection of innovation as the most appropriate form of protection for software. However, as discussed above, in establishing a new legal regime for software protection, the direct protection system needs more development concerning the novelty criterion, the issue of prior art, the scope of protection (e.g. "technical equivalence" or "commercial equivalence") and so on. Complementary aspects of the two proposals by Kronz and Kingston should also be considered and the better elements of each adopted. For example, the criterion of "technical equivalence" appears to be more recommendable than that of "commercial equivalence" as far as it is concerned with the software protection. In terms of the novelty criterion, the criterion of initial commercialization in the innovation patent, which is similar to copyright system, appears to be more appropriate for the software protection than that of purchasability in the innovation warrant. Elimination of the warrant-holder's burden in the innovation warrant is good

for SMEs to be treated equally to large firms with regard to a particular innovation.

7.3 Suggestions for the Introduction of the Direct Protection

Computer programs freely move around the world through the Internet. If innovative software is protected in only one national jurisdiction, that would not help the innovator substantially because the software would be copied in the other jurisdictions without any restriction. Development of the Internet and information technology leads us to consider worldwide protection for software, i.e. unitary innovation patents based on the World Innovation Patent system.[33]

In introducing the Direct Protection of Innovation at an international level, there could be two options: one is to establish a world unified Innovation Office in which a unitary innovation patent is granted; the other is to establish national innovation patent offices in which a unitary innovation patent is filed, published and examined based on World Innovation Patent system. Both of the two options require a *Software Innovation Convention*, which is based on the concepts of the direct protection of innovation, between member countries. A main problem with the latter choice is difficulties in maintaining consistency in the examination in every jurisdiction. It would be extremely difficult to keep an equal level of examination criteria throughout all member countries. Close links between national patent offices would be essential to exchange the information such as prior art, applications, publications or examination results. In order to accomplish a satisfactory cooperation, information networks between national patent offices would be necessary.

If a unified Innovation Office is established, consistency and quality in the

---

[33] 1 Yale Symp. L. & Tech. 3 (1999), *World Patent System Circa 20XX, A.D.* This paper is available at http://lawtech.law.yale.edu/symposium/98/speech_mossinghoff.htm accessed 8 December 2001.

examination could be secured, because standardized education for examiners, establishment of relevant prior art, uniform examination guidelines and so on could easily be accomplished. Uniform examination is very important in software innovation patents, because if an innovation patent is granted for a computer program, it would be valid in all member countries. Thus, the author suggests the following:

- A world unified Innovation Office is established at the WIPO[34] where filings may be made to grant protection in every country based on a *Software Innovation Convention*. According to the *Convention*, a member may file a single application to the Innovation Office and receive protection in each country.[35]
- The Innovation Office deals with main administration such as examination, publication and issuance of innovation patents.
- In order to cooperate with the central Office, local innovation offices are established at each member country's patent office. The local innovation offices would connect between domestic applicants and the central Office. The local offices would receive applications and send them to the central Office. They would also satisfy individual needs of applicants.
- An application can be made by filing either in the Innovation Office directly or in any member country's innovation office which would transfer the application to the central Innovation Office.
- A worldwide search is performed at the Innovation Office.

---

[34] WIPO is a UN agency headquartered in Geneva, established in 1967. As of July 1998, it had 171 members. *Joint WTO-WIPO Press Release*, 21 July 1998. As of January 2002, it had 175 countries. 110 of these have signed the PCT. WIPO, Annual Report 2000, available at www.wipo.org, cited by B. Zorina Khan, *Intellectual Property and Economic Development: Lessons from American and European History*, www.iprcommission.org/documents/Khan_study.pdf accessed 6 June 2002.

[35] Paley, *A Model Software*, 30 January 1996. This article is available at http://members.aol.com/paleymark/ModelAct.htm or http://members.aol.com/paleymark/ModelAct.pdf

- The definition of novelty extends to "not available in the ordinary course of trade within the Community's boundary" or "anywhere in the world" when the Community comprises worldwide countries as the Patent Cooperation Treaty (PCT) does.[36]
- Innovation patents are granted when computer programs are initially commercialized within the Community's boundary (*First-to-Commercialize Priority System*).[37] A useful legal effect, however, is attached to the date of filing in the Innovation Office.[38] When the date of commercialization is not clear, the filing date is adopted.
- The scope of protection is determined by "technical equivalence".
- The enforcement of the innovation patents is publicly rendered by the cooperation between the central Innovation Office and the local innovation offices.
- Once granted, the innovation patent is incontestable unless acquired by fraud.
- For the Innovation Office to work effectively, it is necessary to establish an inventory of computer programs that are used, or have been publicly used in the Community.[39] The digital prior art database would be accessible via the Internet from the local innovation offices or anywhere in the world.
- Examiners of the Innovation Office are composed of experts from the member countries.
- The Innovation Office functions as the software library and licensing agency. Anyone interested in a program should be able to license the program with

---

[36] Kingston, *Direct Protection*, p. 63.
[37] The term, '*First-to-Commercialize Priority System*', is devised by the author in line with the term, first-to-file priority system.
[38] Kingston, *Direct Protection*, p. 263.
[39] Ibid., p. 270.

licensing fee which should be transferred to the program owner.[40] The program owner should pay a certain amount of administration fee to the Innovation Office.

- English is used as the official language. This reduces administration costs and possibility of misunderstanding between applicants and the Innovation Office.[41]
- A world innovation patent court or an arbitration tribunal should be established at the WIPO.
- At the outset, groups of countries may form a Community, a single *domestic* area for innovation patents.[42] Afterwards, the Community may include more countries widely. Leadership toward the world system would be coming from three jurisdictions, i.e. Europe, the US and Japan, by establishing a trilateral innovation patent system covering the three jurisdictions. Europe, the US and Japan would need to negotiate a convention, e.g. *Software Innovation Convention*, to set up the trilateral system. Other countries should be permitted to join this system.

---

[40] Jozef Halbersztadt, *Remarks on the Patentability of Computer Software – History, Status, Developments*. This is available at http://swpat.ffii.org/penmi/linuxtag-2001/jh/indexen.html accessed 8 December 2001.

[41] See http://lawtech.law.yale.edu/symposium/98/speech_mossinghoff.htm. More than 75% of all technical information is published in English first, and almost 90% of all technical information on the Internet is in English.

[42] Kingston, *Direct Protection*, p. 57.

Bibliography

1. Primary Sources

1) Legislation, etc

Europe

EU Commission, *Amended proposal for a Directive on the protection of inventions by utility model*, 7 December 1999.

_____. *Consultations on the impact of the Community utility model in order to update the Green Paper on the Protection of Utility Models in the Single Market* (COM(95)370 final), Brussels, 26 July 2001.

_____. *Consultation Paper on "The patentability of computer-implemented inventions"*, Brussels, 19 October 2000. (This document is available on the Commission's DG Internal Market website at http://europa.eu.int/comm/internal_market/en/intprop/indprop/index.htm).

_____. *Proposal for a Council Regulation on the Community Patent,* Brussels, 1.8.2000 COM (2000) 412 final 2000/0177(CNS).

_____. *Proposal for a Directive of the European Parliament and of the Council on the Patentability of Computer-implemented inventions*, Brussels, 20 February 2002, COM(2002) 92 final, 2002/0047 (COD). Available at http://www.europa.eu.int/comm/internal_market/en/indprop/com02-92en.pdf.

_____. *Proposal for a European Parliament and Council Directive: approximating the legal arrangements for the protection of inventions by utility model*, 16 December 1997.

The Directive on the Legal Protection of Computer Programs – Council Directive 91/250/EEC.

European Patent Convention (EPC).

Guidelines for the Examination in the EPO (Chapter IV Patentability of Part C-Guidelines for Substantive Examination) http://www.european-patent-office.org/legal/gui_lines/e/index.htm accessed 7 April 2001.

US

Business Methods Patent Initiative: An Action Plan, www.uspto.gov/web/offices/com/sol/actionplan.html accessed 6 November 2000.

Business Method Patent Improvement Act of 2000, www.house.gov/boucher/docs/bmpiapage.htm accessed 8 April 2001.

Title 17, US Copyright Act, Section 101-908 (17 U.S.C.).

Title 35, Patents of United States Code (35 U.S.C.).

USPTO, *The Examination Guidelines for Computer-Related Inventions*, 1996 (available at http://www.bitlaw.com/source/soft_pats/final.html accessed 21 January 2001).

USPTO, *Manual of Patent Examining Procedure* (MPEP), August 1993.

USPTO, *A USPTO White Paper: Automated Financial or Management Data Processing Methods (Business Methods)*, 29 March 2000, (This is available at http://www.uspto.gov/web/menu/busmethp/index.html accessed 6 January 2001).

Japan

JPO, *Examination Guidelines For Computer Software-related Inventions*, June 1996.

_____. *Implementing Guidelines for Inventions in Specific Fields*, 1997 (http://www.jpo.go.jp/infoe/sisine.htm accessed 15 December 2001).

_____. *Implementing Guidelines for Industrially Applicable Inventions*, 1997.

_____. *The Guidelines for Examination of Basic Requirements for Utility Model Registration (1993).*

Korea

KIPO, *Industrial Property Laws of The Republic of Korea*, 2000.

_____. *Examination Guidelines for E-commerce Related Inventions*, August 2000.

2) Cases

EPO

T 208/84, 15 July 1986 *Computer-related invention/Vicom* (OJ 1987 [14]).

T 0935/97-3.5.1 of 4 February 1999, [1999] R.P.C. 861 and *T 1173/97-3.5.1* of 1 July 1998, OJ 1999 EPO [609], *IBM* cases.

T 1002/92 OJ 1995 EPO [605], *Queuing system/Petterson.*

T 769/92 (OJ 1995 [525]), The *Sohei* case.

T 0931/95-3.5.1, OJ EPO 10/2001 [441] Technical Boards of Appeal of the EPO, decision of 8 September 2000, *Improved Pension benefits system.*

US

AT&T.Corp. v. Excel Communications, Inc., 172 F.3d 1352 (Fed. Cir.), cert. Denied, 120 S. Ct. 368 (1999)

Cincinnati Traction Co. v. Pope, 210 F. 443 (6th Cir. 1913).

Diamond v. Chakrabarty, 447 U.S. 303 (1980).

Diamond v. Diehr, 450 U.S. 175, 209 USPQ (BNA) 1 (1981).

Gottschalk v. Benson, 409 U.S. 63, 65 (1972), 175 USPQ 673 (1972 US Supreme Court).

Hotel Security Checking Co. v. Lorraine Co (1908).

International News Service v. Associated Press (INS), 248 U.S. 215 (1918).

In re Abele, 684 F.2d 902, 907 (CCPA 1982).

In re Abrams, 188 F.2d at 166.

In re Alappat, 33 F.3d 1526 (Fed. Cir. 1994).

In re Dixon, 44 F.2d 881 (CCPA 1930).

In re Freeman, 573 F.2d 1237, 197 USPQ (BNA) 464 (CCPA 1978).

In re Grams, 888 F.2d 835, 12 USPQ2d (BNA) 1824 (Fed. Cir. 1989).

In re Lowry, 32 F.3d 1579 (Fed. Cir. 1994).

In re Musgrave, 431 F.2d 882 (CCPA 1970).

In re Prater, 415 F.2d 1378 (CCPA 1968).

In re Russell, 48 F.2d 668 (CCPA 1931).

In re Sterling, 70 F.2d 910 (CCPA 1934).

In re Trovato, 42 F.3d 1376 (Fed. Cir. 1994).

In re Walter, 618 F.2d 758, 763 (CCPA 1980).

In re Warmerdam, 33 F.3d 1354 (Fed. Cir., 11 August 1994).

Paine, Webber, Jackson & Curtis, Inc. v. Merril Lynch, Pierce, Fenner & Smith, Inc. 564 F. Supp. 1358, 218 USPQ (BNA) 212 (D.Delaware, 1983).

Parker v. Flook, 473 U.S. 584, 198 USPQ (BNA) 193 (1978).

State Street Bank v. Signature Financial Group, INC 38 USPQ 2d 1530 (D. Mass. 1996), 149 F.3d 1368 (Fed. Cir. 1998).

Japan

Shinpanpyeong 10-5570 Ho. http://www.ipdl.jpo.go.jp/Shinpan/spsogodbk.ipdl.

Supreme Court decision of 30 April 1953, Civil Litigation Precedents Vol. 7 No. 4, p.461.

Tokyo High Court decision of 14 November 1953, Administrative Litigation Precedents Vol. 4 No. 11, p. 2716.

Tokyo High Court decision of 25 December 1956, Administrative Litigation Precedents Vol. 7 No. 12, p. 3157.

3) Treaties, Conventions, etc.

Berne Convention Implementation Act of 1998, www.cni.org/docs/infopols/US.Berne.Convention.html accessed 7 January 2002.

The European Patent Convention (EPC).

The Paris Convention.

The Patent Cooperation Treaty (PCT).

TRIPs Agreement.

2. Secondary Sources

1) Books

Amor, Daniel. *e-Business; (R)EVOLUTION*, Information Publishing Group (Korea), 2000.

BMP Genkyukai, *Business Model Patents*, Tokyo Print, June 2000.

Computer Language Standardization Institute, *The Dictionary of Computer*, Young-Jin Publishing Co, Ltd. 2000.

Cornish, W.R. *Cases and Materials on Intellectual Property*, Third Edition, Sweet & Maxwell, 1999.

Dreyfuss, Rochelle., Diane L Zimmerman and Harry First, eds, *Expanding the Boundaries of Intellectual Property*, Oxford University Press, 2001.

Granstrand, Ove. *The Economics and Management of Intellectual Property*, Edward Elgar Publishing Limited, 1999.

Hanneman, H.W.A.M. *The Patentability of Computer Software*, Kluwer Law and Taxation Publishers, 1985.

Hidetoshi Shibata and Tomohito Ihara, *Business Model Patents Strategy*, Douyou-geijai-shinboshya, May 2000.

Holyoak, Paul. and Jon Torremans, *Intellectual Property Law*, Butterworths, London Endinburgh, Dublin,1998.

Intellectual Property Institute (Japan), *A Study of the Protection of Business Methods*, March 2000.

International Intellectual Property Training Institute, *WIPO Asian Regional Seminar on the Implications of the TRIPs Agreement for Enterprises*, December 1996.

Iwajaki Yas-shi, *Business Model Patents*, Kanki Publishing Company, June 2000.

Kingston, William. *Innovation, Creativity and Law*, Kluwer Academic Publishers, 1990.

_____. *The Political Economy of Innovation*, Martinus Nijhoff Publishers, 1984.

_____. ed, *Direct Protection of Innovation*, Kluwer Academic Publishers, 1987.

Knight, H. Jackson. *Patent Strategy for Researchers and Research Managers*, John Wiley & Sons, 1996.

Korean Industrial Property Office (KIPO), *Guidelines to the Internet-related Inventions*, May 2000.

Landis, John L. *Mechanics of Patent Claim Drafting*, Practising Law Institute, 1978.

Lee, Jong-il. *Patent Law*, Hanbit Intellectual Property Rights Centre (Korea), 1999.

Marett, Paul. *Intellectual Property Law,* London, Sweet & Maxwell, 1996.

Minoru, Sano. *Software and Intellectual Property Rights*, Korean Industrial Property Office, August 1998.

Muir, Ian., Matthias Brandi-Dohrn, and Stephan Gruber, *European Patent Law*, Oxford University Press, 1999.

Oster, Karen Dana Fienberg and Kevin L. Russell, *Chernoff's Federal Circuit Patent Case Digests*, West Group, 1998.

Paterson, Gerald. *The European Patent System (The Law and Practice of the European Patent Convention)*, translated in Korean, Korean Patent Attorney Association, 1998.

Reid, Brian C. *A Practical Guide to Patent Law*, Sweet & Maxwell, 1993.

Rivette, Kevin G. and David Kline, *Rembrandts in the Attic*, Harvard Business School Press, 2000.

Sheldon, Jeffrey G. *How to Write a Patent Application*, Practising Law Institute, November 1996.

Takashi, Nakkajima. *Shocking Business Model Patents,* Sigma-Insight Com Ltd., July 2000.

Takenaka, Toshiko. *Interpreting Patent Claims: The United States, Germany and Japan*, VCH Verlagsgesellshaft mbH, D-69469 Weinheim (Germany), 1995.

Tritton, Guy. *Intellectual Property in Europe*, London, Sweet & Maxwell, 1996.

Yoshifuji and Ken'ichi Kumagai, *Patent Law*, Daekwang-Seolim Publishing, June 2000.

2) Articles

Adams, John., Puay Tang and Daniel, *Patent protection of computer programmes*, ECSC-EC-EAEC Brussels-Luxembourg, (University of Sussex at Brighton) 2001.

Aharonian, Greg. *Patent Examination System Is Intellectually Corrupt*, http://swpat.ffii.org/vreji/minra/www.bustpatents.com/corrupt.htm accessed 6 June 2001.

Bagley, Margo A. *Internet Business Model Patents: Obvious By Analogy*, 7 Mich. Telecomm. Tech. L. Rev. 253 (2001), http://www.mttlr.org/volseen/bagley.html.

Basinski, Erwin J. *Business Method Patents in Europe: A Saussurean Explanation*, World E-Commerce & IP Report, April 2001.

Bessen, James. and Eric Maskin, *Sequential Innovation, Patents, and Imitation*, Working Paper Department of Economics, Massachusetts Institute of Technology, No. 00-01, January 2000.

Bitzer, Jurgen. *The computer software Industry in East and West: Do Eastern European Countries Need a Specific Science and Technology Policy?* http://www.diw-berlin.de 13 October 2001.

Chandler, James P. *Patent protection of computer programs*, http://mipr.umn.edu/archive/articles/Chandler2000_01_01.htm accessed 7 January 2001.

Chiappetta, Vincent. *Defining the Proper Scope of Internet Patents: If We Don't Know Where We Want to Go, We're Unlikely to Get There*, 7 Mich. Telecomm. Tech. L. Rev. 289 (2001), available at http://www.mttlr.org/volseven/chiapetta.html.

Cohen, Julie E. & Mark A. Lemley, *Patent Scope and Innovation in the Software Industry*, California Law Review, Vol. 89:1, 2001 www.law.georgetown.edu/faculty/jec/softwarepatentscope.pdf accessed 14 October 2001.

Cohen, Seth A. *To Innovate or Not to Innovate, That is the Question: The Functions, Failures, and Foibles of the Reward Function Theory of Patent Law in Relation to Computer Software Platforms*, 5 Mich. Telecomm. Tech.L.Rev.1(1999). Available at www.mttlr.org/volfive/cohen.pdf (accessed 14 October 2001).

Dam, Kenneth W. *Intellectual Property in an Age of Software and Biotechnology*, Chicago Working Paper in Law & Economics, May 1995.

Davis, Steven J., Jack MacCrisken and Kevin M. Murphy, *Economic Perspectives on Software Design: PC Operating Systems and Platforms*, Working Paper 8411, August 2001. http://www.nber.org/papers/w8411 accessed 14 October 2001.

_____. *The Evolution of the PC Operating System: An Economic Analysis of Software Design*, 29 June 1999, (http://www.nber.org/papers).

Derrick, Douglas C., *It Doesn't Fit: The Dilemma of Computer Software and Patent/Copyright Law*, E Law – Murdoch University Electronic Journal of Law, Vol 3, No 1 (May 1996). www.murdoch.edu.au/elaw/issues/v3n1/derrick.html accessed 1 June

2002.

Dreyfuss, Rochelle Cooper. *Are Business Method Patents Bad for Business?* Santa Clara Computer and High Technology Law Journal, Vol. 16(2), March 2000. This paper also available at: http://papers.ssrn.com/paper.taf?abstract_id=219574.

Economides, Nicholas. *The Microsoft Antitrust Case*, New York University, Center for Law and Business, Working Paper #CLB-01-003, 2 April 2001, (this paper is available at http://papers.ssrn.com/paper.taf?abstract_id=253083).

_____. *Durable Goods Monopoly with Network Externalities with Application to the PC Operating Systems Market*, Quarterly Journal of Electronic Commerce, Vol. 1, No.3 (2000).

Einarsson, Peter. and Marie Bystrom, *TRIPS: Consequences for developing countries Implications for Swedish developing cooperation*, Consultancy Report to the Swedish International Development Cooperation Agency, August 2001, available at www.grain.org/docs/sida-trips-2001-en.pdf accessed 10 June 2002

ETAN Expert Working Group, *Strategic dimensions of Intellectual Property Rights in the context of S&T Policy,* Report prepared for the European Commission, June 1999, p. X, http://europa.eu.int/comm/research/era/pdf/ipr-expertgroupreport.pdf 10 June 2002.

Fagin, Barry. *Standardization/Innovation Tradeoffs in Computing: Implications for High-Tech Antitrust Policy*, www.faginfamily.net/barry/Papers/tradeoffs.htm accessed 15 October 2001.

Fellas, J. *The Patentability of Software-related Inventions in the United States.* European Intellectual Property Review, 1999.

Ginsburg, Jane C., *Four Reasons and a Paradox: The Manifest Superiority of Copyright Over Sui Generis Protection of Computer Software*, 94 Colum. L. Rev. 2559 (Dec. 1994). Available at www.law.cornell.edu/commentary/intelpro/gns94txt.htm accessed 1 June 2002.

Halbersztadt, Jozef. *Remarks on the Patentability of Computer Software – History, Status, Developments.* http://swpat.ffii.org/penmi/linuxtag-2001/jh/indexen.html accessed 8 December 2001.

Hart, Robert., Peter Holmes and John Reid, Study Contract ETD/99/B5-3000/E/106: *The Economic Impact of Patentability of Computer Programs*, Report to the European Commission, 1999.

Hirashima, Ryuta. *Changes in Subject-matter under the US Patent Law*, Institute of Intellectual Property, March, 2000.

Holt, Chris. *Patentability of Internet Business Models*, www.ukans.edu/~cybermom/CLJ/holt.html accessed 7 January 2001.

Jaffe, Adams., Joshua Lerner, and Scott Stern, eds., *Innovation Policy and the Economy*, Volume I, MIT Press, 2001. available at http://haas.berkeley.edu/~shapiro/thicket.pdf.

Jean-Paul Smets-Solanes, *Stimulating competition and innovation in the information*

society, www.pro-innovation.org 23 March 2001.

Kaplow, Louis. and Steven Shavell, *Property Rules Versus Liability Rules: An Economic Analysis*, 109 Harv. L. Rev. 713, 716 (1996).

Karjala, Dennis S. *Copyright, Computer Software, and the New Protectionism*, 28 Jurimetrics Journal 33 (Autumn 1987).

_____. *The Relative Roles of Patent and Copyright in the Protection of Computer Programs*, 17 John Marshall Journal of Computer and Information Law 41 (Autumn 1998).

Khan, B. Zorina. *Intellectual Property and Economic Development: Lessons from American and European History*, Commission on Intellectual Property Rights, Department of Economics and National Bureau of Economic Research, www.iprcommission.org/documents/Khan_study.pdf accessed 6 June 2002.

Kingston, William. *Meeting Nelson's Concerns about Intellectual Property*, http://www.business.auc.dk/druid/conferences/nw/paper1/kingston.pdf accessed 13 August 2001.

Kobak, James B. Jr., *Intellectual Property, Competition Law and Hidden Choices Between Original and Sequential Innovation*, Virginia Journal of Law and Technology, 3Va. J.L. & Tech. 6 (Autumn 1998) http://www.student.virginia.edu/~vjolt/graphics/vol3/vol3_art6.html accessed 22 July 2001.

Kogut, Bruce. and Anca Meitu, *The Emergence of E-Innovation: Insights from Open Source Software Development*, A Working Paper of the Reginald H. Jones Center, WP00-11, The Wharton School, University of Pennsylvania, http://jonescenter.wharton.upenn.edu/events/software.pdf accessed 10 November 2001.

Laakkonen, Ari. *European and UK Software and Business Method Patents are in a Holding Pattern*, World E-commerce & IP Report, Volume 1, April 2001.

Lea, Gary. *The Revolution That Never Was: A Cynic's Eye View of the Software, Business and E-Commerce Method Patenting Controversy in the Wake of State Street*, Digital Technology Law Journal, Volume 2 Number 1 http://wwwlaw/dtlj/articles/vol2_1/leaDTLJ2_1.pdf.

Leith P, *Software Utility Models and SMEs*, 2000 (2) The Journal of Information, Law and Technology (JILT). http://elj.warwick.ac.uk/jilt/00-2/leith.html accessed 3 June 2002.

Lemley, Mark A. and David W. O'Brien, *Encouraging Software Reuse*, 49 Stan. L. Rev. 255, 259-68 (1997).

Lerner, Josh. *Did Microsoft Deter Software Innovation?* http://papers.ssrn.com/sol3/papers.cfm?abstract_id=269498 accessed 16 October 2001.

Lerner, Josh. and Samuel Kortum, *Stronger Protection or Technological Revolution: What is behind the recent surge in patenting?* Working paper 6204.

Maskus. Keith E., *Intellectual Property Rights and Economic Development*, this is available at www.colorado.edu/Economics/mcguire/workingpapers/cwrurev.doc accessed 9 June 2002.

McDonald, Shawn. *Patenting Floppy Disks, or How the Federal Circuit's Acquiescence has Filled the Void Left by Legislative Inaction*, 3 Va. J.L. & Tech. 9 (Fall 1998) 1522-1687 / © 1998 Virginia Journal of Law and Technology Association http://vjolt.student.virginia.edu/graphics/vol3/vol3_art9.html accessed 14 December 2001.

Merges, Robert P. *As Many As Six Impossible Patent Before Breakfast: Property Right for Business Concepts and Patents System Reform*, Berkeley Technology Law Journal, [Vol. 14:577, 1999] www.law.berkeley.edu/journals accessed 6 November 2000.

_____. *Uncertainty and the Standard of Patentability*, http://www.law.berkeley.edu/journals/btlj/articles/07_1/Merges/.../text.htm accessed 30 October 2001.

Merges, Robert P. and Richard R. Nelson, *On the Complex Economics of Patent Scope*, http://cyber.law.harvard.edu/ipcoop/90merg2.html accessed 4 June 2001.

Nordhaus, William D. *The Progress of Computing*, Cowles Foundation Discussion Paper No. 1324, September 2001. Available at Social Science Research Network Electronic Paper Collection: http://papers.ssrn.com/abstract=285168.

Office of Fair Trading, *E-commerce and its Implications for Competition Policy*, Discussion Paper 1, A report compiled by the Frontier Economics Group for the Office of Fair Trading, OFT308, August 2000.

Paley, Mark Aaron. *A Model Software Petite Patent Act*, George Washington University National Law Center, 1996. http://members.aol.com/paleymark/index.htm accessed 1 June 2001.

PbT Consultants Ltd, *The Results of the European Commission Consultation Exercise on The Patentability Computer Implemented Inventions,* Contract number PRS/ 2000/ A0-7002/E/98. http://europa.eu.int/comm/internal_market/en/indprop/softanalyse.pdf accessed 14 October 2001.

Rai, Arti K. *Addressing the Patent Gold Rush: The Role of Deference to PTO Patent Denials*, University of San Diego, School of Law, Public Law and Legal Theory Working Paper No. 05, February 2000.

Rasiah, Rajah. *TRIPs and Capability Building in Developing Economies*, March 2002, United Nations University, Institute for New Technologies, Discussion Paper Series 2002-1.

Sabbatini, Pierluigi. *The Microsoft Case*, available at the Social Science Research Network Electronic Paper Collection: http://papers.ssrn.com/paper.taf?abstract_id=223769 accessed 14 October 2001.

Sakakibara, Mariko. and Lee Branstetter, *Do Stronger Patents Induce More Innovation? Evidence from the 1998 Japanese Patent Law Reforms*, NBER Working paper

7066.

Samuelson, Pamela., Randall Davis, Mitchell D. Kapor, and J.H. Reichman, *A Manifesto Concerning The Legal Protection of Computer Programs*, 94 Colum. L. Rev. 2308 (1994).

Samuelson, Pamela. and J.H. Reichman, *Intellectual Property Rights in Data?* Vanderbilt Law Review, January 1997. 50 Vand. L. Rev. 51. http://eon.law.harvard.edu/h2o/property/alternatives/reichman.html, accessed 21 July 2001.

Schumm, Brooke III. *Escaping the World of "I Know It When I See It": A New Test For Software Patentability*, 2 Mich. Telecomm. Tech. L. Rev. 1 (1996) available at http://www.mttlr.org/voltwo/shumm.pdf.

Sidak, J. Gregory. *An Antitrust Rule for Software Integration*, Yale Journal on Regulation, 2001 (This paper is available at http://SSRN_ID268508_code010430130.pdf accessed 13 October 2001).

Witek, Keith E. *Developing a Comprehensive Software Claim Drafting Strategy for U.S. Software Patents*, http://www.law.berkeley.edu/journals/btlj/articles/11_2/Witek/html/text.html, accessed 21 April 2001.

3) Others

The EU Committee, *The EU Committee's Response to the Commission's Consultation Paper on Patents for Computer Implemented Inventions Initial Discussion*, 12 January 2001, (This document is available at the EU Committee website: http://www.eucommittee.be).

Intellectual Property Institute (Japan), *Report on Protection Method of New Field (Business Methods)*, March 2000.

Japan Patent Office (JPO), Asia-Pacific Industrial Property Center, JIII, Ken-ichi KUMAGAI, Faculty of Law, Kyusyu University, *Outline of Utility Model System*, 1999.

JPO, Asia-Pacific Industrial Property Center, JIII, *Practical procedures for prosecuting software patents*, 1999 www.apic.jiii.or.jp/facility/text/1-03 accessed 22 September 2001.

JPO, *Pro-patent Era in Japan*, February 1997.

JPO, The Planning Subcommittee of the Industrial Property Council, *Report of the Planning Subcommittee of the Industrial Property Council-To the better understanding of pro-patent policy*, November 1998. http://www.jpo-miti.go.jp/indexj.htm accessed 13 March 2001.

JPO, *Towards the International Harmonization of Intellectual Property Rights Systems in the 21st Century*, www.jpo-miti.go.jp/tousie/chapter1.htm accessed 20 June 2000.

Kotler, Mindy L. Gary W.Hamilton, Esq. *A Guide to Japan's Patent System,* US Department of Commerce Office of Technology Policy, Asia-Pacific Technology Program, November 1995.

Office of Fair Trading, *E-commerce and its Implications for Competition Policy*, Discussion Paper 1, a report compiled by the Frontier Economics Group for the Office of Fair Trading, OFT308, August 2000.

The UK Patent Office, *Should Patents be Granted for Computer Software or Ways of Doing Business?* http://www.patent.gov.uk/about/consultations/conclusions.htm accessed 4 May 2001.

US Federal Trade Commission Staff, *Anticipating the 21st Century: Competition Policy in the New High-Tech, Global Marketplace*, Volume I, May 1996.

WIPO, *WIPO Primer on E-Commerce and IP Issues*, May 2000.

4) Internet sites

http://brie.berkeley.edu/~briewww/pubs/wp/wp79.pdf accessed 13 October 2001.
Islands in the Bit-Stream: Charting the NII Interoperability Debate, Francois Bar, Michael Borrus and Richard Steinberg, Berkeley Roundtable on the International Economy (BRIE), University of California, Berkeley, BRIE Working Paper 79, 1995.

http://eplu.netgistics.com/PDF/1999-2000/Fall/BUSA541/cham1.pdf accessed 14 October 2001.

http://europa.eu.int/comm/internal_market/en/intprop/indprop/studyintro.htm accessed 4 May 2001.

http://europa.eu.int/comm/internal_market/en/intprop/indprop/utility.htm accessed 25 June 2001.

http://info.sm.umist.ac.uk/esrcip/Projects/L5253004/final_report.htm accessed 8 October 2001.

http://isu.indstate.edu/welsh/ua/hist-comp.html accessed 6 October 2001.

http://jeffsutherland.com/papers/Rans/OOlanguages.pdf accessed 14 October 2001.

http://lawtech.law.yale.edu/symposium/98/speech_mossinghoff.htm accessed 8 December 2001.

http://lookup.computerlanguage.com/host_app/search accessed 16 October 2001.

http://math.ucsd.edu/~fan/rep.pdf accessed 26 October 2001.

http://papers.ssrn.com accessed 10 February 2000.

http://papers.ssrn.com/sol3/papers.cfm?abstract_id=239903 accessed 16 October 2001.

http://searchsecurity.techtarget.com/sDefinition/0,,sid14_gci212062,00.html accessed 16 October 2001.

http://www-2.cs.cmu.edu/~amulet/papers/uihistory.tr.html accessed 10 October 2001.

http://www.bustpatents.com/aharonian/softcopy.htm accessed 29 January 2002.

http://www.jpo.go.jp/tousie/pdf/bukai_report_e.pdf accessed 10 June 2002. *Report Presented by the Intellectual Property Committee of the Industrial Structure Council.*

www.computerlanguage.com/sitemain/content.html accessed 16 October 2001.

www.computer.org/students/looking/spring97/janlee/ accessed 6 October 2001.

www.developer.ibm.com/library/articles/schenk1.html *Linux: Its history and current distributions* accessed 8 January 2002.

www.dit.upm.es/~joaguin/report_en.pdf accessed 13 October 2001, *Software patents and their impact in Europe.*

www.diw-berlin.de accessed 13 October 2001.

www.digitalcentury.com/encyclo/update/com_hd.html accessed 30 October 2001, *Computers: History and Development.*

www.Eurolinux.org accessed 13 August 2000.

www.europa.eu.int/comm/internalmarket/en/intprop/indprop/softpaten.htm accessed 26 October 2000.

www.europa.eu.int/comm/internal_market/en/indprop/com02-92en.pdf accessed 20 February 2002.

www.faginfamily.net/barry/Papers/tradeoffs.htm accessed 15 October 2001.

www.ipmall.fplc.edu/hosted_resources/jorda_08_27_99.htm accessed 7 June 2002. WIPO-UNITAR Academy, New York City, August 26-27, 1999.

www.iprcommission.org/documents/Khan_stuty.pdf accessed 6 June 2002. B.Zorina Khan, *Intellectual Property and Economic Development: Lessons from American and European History.*

www.jpo-mini.go.jp/index accessed 17 February 2001.

www.jpo-mini.go.jp/indexj.htm accessed 13 March 2001.

www.jpo-miti.go.jp/saikine/ accessed 9 February 2000; *Report on Comparative Study Carried out under Trilateral Project 24.2.*

www.ladas.com/GUIDES/COMPUTER/Patents.USA.html accessed 7 January 2001, *Ladas & Parry Guide to Statutory Protection for Computer Software in the United States.*

www.law.asu.edu/HomePages/Karjala/Articles/JurimetricsFall1987.html#FN;Fa accessed 21 August 2001. Dennis S. Karjala, *Copyright, Computer Software, and the New Protectionism,* 28 Jurimetrics Journal 33 (Autumn 1987).

www.law.berkeley.edu/institutes/bclt/pubs/ipnta/appenb.pdf accessed 13 October 2001.

www.nber.org/papers accessed 14 October 2001.

www.nesl.edu/lawrev/vol32/vol32-3/chan.htm accessed 14 January 2001, *The*

Patentability of Software Data Structures after Lowry and Warmerdam.

www.nihonlinks.com/JamesMiller/OpenSourceMoralRights/Draft-0310.pdf accessed 14 October 2001.

www.patent.gov.uk/about/consultations/annexa.htm accessed 4 May 2001.

www.patent.gov.uk/about/consultations/conclusions.htm accessed 11 May 2001, UK Patent Office, *Conclusions to the Consultation on "Should Patents be Granted for Computer Software or Ways of Doing Business?"*

www.princeton.edu/~mike/articles/hcht/hcht.html accessed 6 October 2001. Michael S. Mahoney, *The History of Computing in the History of Technology*, Annals of the History of Computing 10 (1988), 113-125.

www.science.uva.nl/faculteit/museum/technotrends.html accessed 6 October 2001.

www.southcentre.org/publications/trips/tripsmaintexttrans-01.htm accessed 5 June 2002. *TRIPs – Trade-Related Intellectual Property Rights.*

www.udayton.edu/~lawtech/papers accessed 20 June 2001.

www.uspto.gov/web/menu/busmethp/index.html accessed 5 January 2002.

www.uspto.gov/web/offices/com/sol/actionplan.html accessed 18 October 2000.

www.win.tue.nl/xootic/magazine/mar-2000/vanommering.pdf accessed 14 October 2001. Rob van Ommering, *A Composable Software Architecture for Consumer Electronics Products*, Xootic Magazine, (March 2000).

www.worldbank.org/propects/gep2002/chapter5.pdf accessed 6 June 2002. *Intellectual Property: Balancing Incentives with Competitive Access.*