# The Patentability of a Computer Program as a Function of Its Relational Characteristic with Hardware

Sang Mu Lee

## CONTENTS



## ABSTRACT

The patentability of computer program has been discussed because of its deviation from the traditional definition of a patent. The relativities of computer programs to hardware are classified to measure the relative patentability of computer programs in this paper. It can be seen through the patentability analysis that the change in patentability basically follows an exponential function of the hardware character of the software, and the coefficient of an exponent part of the function is a damping factor that determines a patentability degree or trend. The basic patentability of computer programs is revealed when the damping factor value is 1, and a statistical patentability trend is derived. In drafting a patent specification, an appropriate expression of applicability and substantiality of computer programs is needed to acquire a patent right.

# I. INTRODUCTION

Human beings possess property for their social lives. Generally the property is tangible and material. However, as an original thought or an idea is based on creative human mental activity, not on a concept of materiality, what is defined so that legal rights and interests may be protected is intellectual property rights. Out of these intellectual property rights, a representative right applicable to industrial fields is a patent.

One of the industrial fields which is highlighted and has been developed these days is the computer field. The primary categories of this field have been classified into hardware and software since it developed to some degree from a basic calculator. A patent system protects an inventer's exclusive rights and interests for an original idea applicable to an industrial technology. Thus an original develoment of a computer-implemented technique may be the patentable subject matter. From the special viewpoint of a patent system, computer hardware does not vary from the traditional basic nature of patent, but computer software alone is different because software is a meaningless thing by itself and contrary to the subject matter patentable under U.S. patent law. It thus has been questioned how a patent application for an original idea of software for diverse domains of the computer field should be examined.

The basic articles on this problem are from the U.S., and they present advanced guidelines for considering software patentability mainly from the statutory viewpoint [1]-[3]. Those guidelines affect the international trend of patent grants for software. However, even though case-oriented decisions have been made, systematic theory from the fundamental meaning of software and patents is not established, and thus problems remain to be solved for practical application of guidelines.

In this situation, the relativity of a computer program to hardware through which computer programs are substantialized is analyzed in this paper to illuminate the intrinsic character of computer program patentability. This paper is regarded as a new attempt of analysis and approach in that it applies an engineering method and provides a statistical result.

It defines the patentability of a computer program by manifesting the relation between software and patentabiliby through a qualitative/quantitative analysis and application of statistics. In addition, a technique to enhance the patentability of a computer program is suggested.

# II. PREMISES

## 1. Historical Background

In the past, the USPTO[4] did not easily acknowledge the patentability of a computer program. Instead, it regarded a computer program as a thing like a mathematical algorithm or a logical procedure corresponding to an abstract idea or natural

[4]United States Patent and Trademark Office

law itself which is not patentable. After the CCPA[5] judged that the patentability of a computer program should be admitted by the warehouse theory,[6] they decided that a patent should be given for an invention of a certain kind of program by changing the former policy [4].

There have been several representative cases recently which define the change in the criterion of patentability for a computer program, and they are as follows. The ALAPPAT case (FED.CIR.1994) reconfirmed that everything made by man under the sun except natural law, a natural phenomenon, and abstract ideas may be patentable subject matter as prescribed in the Article 101 of the U.S. patent law. The LOWRY case (FED.CIR.1994) held that a computer memory having a data structure could be patentable subject matter if it had a new feature in the composition of the data stucture. The BEAUREGARD case held that a computer program recorded on tangible media, for example, a floppy diskette, was patentable subject matter [5].

In line with these series of events, Japan is also proceeding to legalize a patent right for the software recorded on a storage media such as a floppy diskette or a CD-ROM, etc. This changes the existing law which has recognized a patent right only for computer software unified with an apparatus such as machinery and tools. Japan is studying this issue so as to solve every kind of relevant problem about this.

In Korea also, we have established an offical group in charge of computer inspection in the 4th Inspection Bureau of the Patent Office, and are concentrating efforts on social activities for research concerning computer software.

## 2. Viewpoint of Formation Nature[4]

The basic issue of discussion on the patentability of a computer program is whether a computer program has the character required to constitute statutory subject matter as defined in the Title 35 U.S.C.101, and therefore the patentability of a computer program is examined from that viewpoint in this paper.

A patent is given for an implementation of an original idea, and patentable subject should not be abstact or ideal but be an entity having a real specific use and purpose. The patentable classes of statutory subject matter are process, machine, manufacture, and composition of matter and new uses and improvements thereof [6].

Korea and Japan basically distinguish the patentability of computer programs from the usability of natural law. Natural law by itself is not patentable, but an application using it is patentable. The usability of natural law may be regarded as meaning patents are applicable just to industrial

[5]Court of Customs and Patent Appeals

[6]This theory authorizes the patentability of a computer program as a part of device

[4]The Formation Nature is defined from the Article 101 of U.S. Patent Law and the usability of natural law in Korea and Japan's patent law

fields. Its interpretation with respect to a certain computer program may be hard to predict. Natural law is an already existing phenomenon, and a patent must be a new entity using it.

### 3. Class Nature of Computer Program

The class nature of a computer program, with respect to patentability, means that computer programs differ in degree of closeness or interaction with hardware to some degree. In accordance with this class nature, computer programs can be largely classified as in Fig. 1.

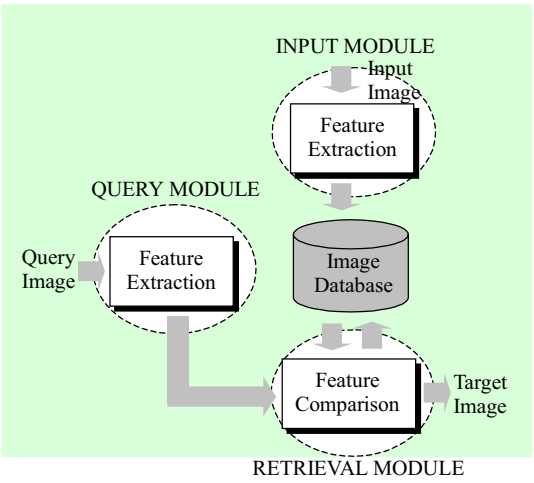


**Fig. 42.** Class nature of compter programs.

There are two main categories of computer programs: "application" programs and "operating system" programs. Application programs usually perform a specific task for the computer user, such as word processing, checkbook balancing, or game playing. Application programs are programs that directly interact with the computer user.

By contrast, operating system programs interact more with the computer than with the computer user. Operating system programs generally manage the internal functions of the computer or facilitate use of application programs. The operating system is the brain of the computer. It determines the computer's capability to store in memory, operate items such as view terminals, and do other basic operating chores [7].

Figure 1 explains the class nature of computer programs. At the center of the circles is hardware (H/W). Above it, is firmware (F/W) by microcode programming. Operating system (O/S) and application programs (A.P.) are located next in order. This classification is a rough one, and more detailed classifications may be considered.

## III. PATENTABILITY ANALYSIS OF COMPUTER PROGRAMS

### 1. Substantiality and Natural Law Usability

The usability of natural law appears through relevance to hardware. Basically, computer programs operate only in combination with hardware. The relativity of a computer program to hardware has two types largely. Some types of computer programs control, activate, or operate hardware while some others just use or need it

for program execution purposes. In the former case, the substantiality, that is, the usability of natural law is easy to distinguish. However, it is not easy to express the usability of natural law in the latter case where the applicability of the user side is strong.

May an original progression method of a game program as such an application program be regarded as an idea to use natural law? For example, there is a flight game program of universe flight, Novastorm (Psygnosis, Ltd., 1994). In the basic progression pattern of this game, a fighter plane must destroy numerous enemy planes by checking for arrival at main objective points. Destroying enemy planes generates tokens. And to collect these tokens, a bomb carried by the fighter plane is reinforced or additional weapons can be acquired.

In the progression of this game, there may be a difference from the viewpoint of its relation with a device or the usability of natural law which can be called the substantiality. One thing is the relation with natural law appearing purely in the progression procedure itself on the screen. Another is the relation with computer hardware components related to the program progression regardless of the program character. First, the motion itself is activity based on natural law from the viewpoint of the pure progression procedure. This is motion in an imaginary actuality on the screen, so it is not the substantial use of natural law. However, the motion is formed by the functions of computer hardware components. That is, a central processing unit in the computer controls the background screen on the monitor device, moves substances, generates a sound effect through a speaker linked to the computer and forms the motions in response to the terminal contacts of a keyboard by loading the main program into the main memory unit from the auxiliary memory, reading this in order, and executing the coded commands. These successive motions are by action with hardware, which is common in all computer programs. The motions of a program are basically through hardware devices in any form. It is regarded as a difference in interpretation or viewpoint whether such motions have the direct influence of natural law or they have only an indirect relation. The framework of hardware related to natural laws or substantiality bases the procedure of computer programs. This is similar to a man composed of bones to contain internal organs and to support the flesh, a brain corresponding to the central processing unit of computer to move them, and invisible mental thought corresponding to software of a computer taking charge of the substantial movement of these structures.

## 2. Potential Patentability of Computer Programs

We need to exclude the fixed thinking that a computer program is made of human logical thought and is a mathematical algorithm to understand the potential patentability of computer programs. We do not regard invisible computer programs

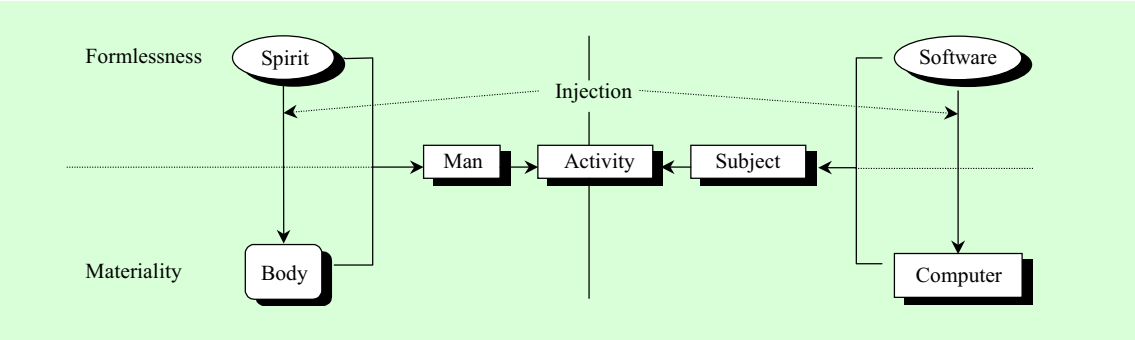


**Fig. 43.** Potential patentability of computer program.

alone as the subject matter of patent. A computer program has significance only in that it was named as a certain special means used in a computer. One viewpoint from which to appreciate its patentability is to notice whether a substantial effect appears when the program unites with a computer. A program and a computer can not have meaning anyway when they are separated individually. And combined, they are regarded as one entity and the subject matter of a potential patent. Whether a computer program uses the natural law directly or as far as it is not a natural phenomenon itself or abstract idea [5], there is only a difference in degree, but it may be deemed that it has patentability.

As a man is formed with a body and a soul, a computer is formed with hardware and software. So any one thing cannot exist independently. And through the combination of both, one independent individual domain is formed from that time.

Software by itself has only mental character. However, it comes to appear as a substantial existence after being infused into hardware. And it becomes the power source of hardware to move by using natural law. And human beings attain the goal that they seek by using these resources. Thus, basically, in a computer program (i.e. software) it is deemed that patentability is dormant.

Figure 2 explains this situation where the center vertical line is a division between the left side which is a human section and the right side which is a computer. It is symmetrical. Also along the horizontal center dashed line, the upper half is formlessness and the bottom half is materiality.

As a man is made by injection of spirit to a body, a computer is made by injection of software into hardware. Thereby, both entities come to act. Additionally a computer designates only hardware or what is activated by software. A computer as subject is the latter case. A man or a computer operates in natural law and appears to be an acting subject in real life. In this meaning, the ultimate summit of computer technology may be the manufacture of a man-made man through the development of artificial intelligence.

## 3. Qualitative Analysis

All computer programs have potential patentability basically as described in the above explanation. Potential patentability appears when computer programs act on computer hardware. The computer program is formless by itself, and its functions are substantiated when related to computer hardware.

The fact that a computer program materializes only with hardware means that a computer program is related to patentable subject matter in the U.S. patent law or natural law usability in Korea and Japan patent law because substantiality or the natural law usability appears through real matter. By the way, the patentability of computer programs changes according to the kind of computer program.

Computer programs have class nature according to their relation to hardware. That is, there is a character change of computer programs according to their interaction with hardware. There are programs which are closely or directly related to hardware or have strong applicability oriented to a computer user's convenience. Software nature is defined here as a degree ranging far from hardware level to pure software level. Hardware nature can be inversely defined as its degree of closeness to hardware.

When a subject does not have software nature at all, it is hardware itself having no problem in patentability from the viewpoint of subject matter or natural law usability among patentable requisites. There is no problem in patentability from the viewpoint of subject matter or natural law usability among patentable requisites is defined as the formation nature. On the other hand as it becomes a computer program itself when software nature approaches infinity, patentability disappears. Figure 3 diagrams this quality briefly.

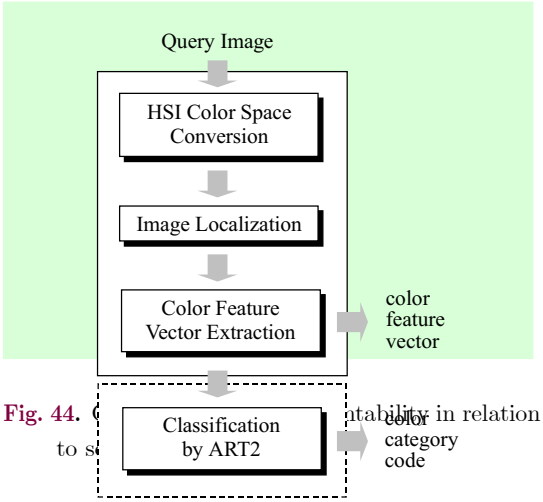


Fig. 44. C ntability in relation to s

From the above qualitative analysis, the basic theory is induced that the patentability of computer program according to its software nature follows an exponential function. When the software nature, S, is zero, the patentability, P, is '1'. That is, patentability is 100% because the subject matter from the viewpoint of formation nature in that case is hardware itself. Therefore the coefficient of the exponential function causes the function value to become '1' (i.e., the exponent becomes zero). Software nature cannot be less than zero. When the 'S' goes to the infinity, 'P' goes to zero. So the sign of the exponent variable 'S' is minus.

The general function for patentability by the above theory can be represented by

$$P = e^{-dS} \ (S \geq 0). \tag{1}$$

In (1), the base of the exponential function here takes the Napier number, 'e' as being representative [8]. The notation 'd' of the exponent part means that the coefficient determines the damping rates of the exponential function.

## 4. Quantitative Analysis

### A. Analysis Method

In quantitative analysis, there is difficulty in correlating the engineering character of a computer program and the statutory character of patentability. In the transfer relations for an input and output function which has software nature (S) as input and patentability (P) as output, the application procedure for each variable is depicted in Fig. 4.

The S variable is defined as discrete values. It is graded according to the degree of relationship of a computer program to hardware. The relativity types of computer programs to hardware are classified and computer programs are sorted into types of relativity for gradation. Patent statistics of computer programs according to their software nature is applied as an outside element from the standpoint of probability instead of intrinsic change of patentability because the variables here are not physical in character in which experimental data can be otherwise obtained.

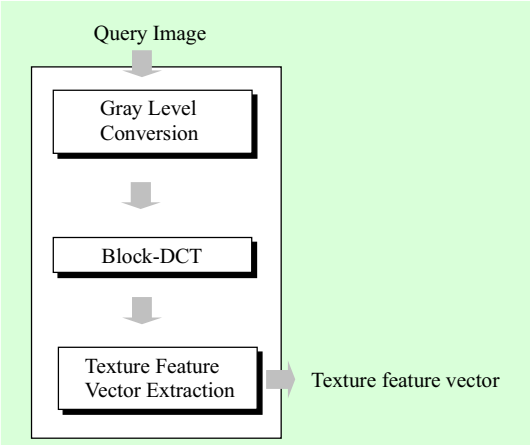


**Fig. 45.** Quantitative analysis procedure.

### B. Classification of Computer Programs

Table 1 classifies relativity types of computer programs with respect to hardware and applies general computer programs to the types. The grade expresses degree of closeness of corresponding computer programs to hardware level or the strength of software nature. The higher the value of a grade is, the stronger the software nature is. The level grade of hardware itself is '0' and the grade of software itself is infinity. Firmware is microcoded in internal devices of a computer to define its operation. It sticks closest to hardware, and is almost hardware itself. Microcoding is also programming which determines internal operations of devices constituting a computer. Therefore firmware is classified into software in a wide variety of ways [7]. It is classified into grade 1 just next to hardware itself.

**Table 9.** Classification of computer programs by hardware relativity.

| Relativity types | Related programs | Grade |
|---|---|---|
| ☐ Hardware itself | - | 0 |
| ☐ Micro-operation | Firmware | 1 |
| ☐ Direct control of hardware and components in compu- ter or peripherals for use | Operating system (and any control programs) | 2 |
| ☐ Access to computer elements for supplementary use | Utility | 3 |
| ☐ Signal flow between computer and external elements | Interface | 4 |
| ☐ Concentrated use of specific components of computer | | 5 |
| - Use of memory and display | GUI[5] (Window program), Graphic, Program development tool | |
| - Use of memory | Expert, DBMS | |
| ☐ Data transfer from computer to computer or other re- mote devices | | |
| - Access to connecting resources | Network | 6 |
| - Information interchange | Communication | 7 |
| ☐ General use of hardware resources | | |
| - Simple need of hardware for operation of software or | Language/Compiler | 8 |
| operation on personal computer purely for user convenience | Word processor | 9 |
| | Game | 10 |
| | Simulation | 11 |
| | Spreadsheet | 12 |
| | Calculation | 13 |
| | Statistics | 14 |
| ☐ Indirect effect on operation | Data structure | 15 |
| ☐ Software itself | - | $\infty$ |

[5]Graphical User Interface

As the operating system is obviously the program which controls and activates computer system hardware constituent for computer use, it comes to grade 2.

In fact, grading is manifest and easy down to operating system level, but difficulty is in application program levels because application programs have some connectivity and duplicity in functional character. Therefore, classification by a completely defined standard is practically impossible. Table 1 includes the main functions of representative computer programs. Detail kinds of programs are included in

some related programs.

The relativity types can be divided into 3 items largely by application areas. The one thing is 'in the computer.' This represents operations in internal or for subsidiary devices of a computer. This can be divided into 5 items again. The first is direct control of components. Grades 1 and 2 correspond to this. The second is access to computer elements for supplementary use. This is grade 3 of utility including diagnostic, test, debugging, security, vaccine, emulation program, etc. Utility programs are allowed to be part of an operating system [9]. The third is concentrated use of specific resources. Grade 5 corresponds to this. The fourth is general or mere use of hardware. Grades 8 to 14 correspond to the fourth item. The fifth is the indirect effect on operation. Grade 15 corresponds to this item. Data structure is not a program in effect but belongs to software category, and appeared as a new patentable subject matter only recently [10]. The second application area for hardware relativity is 'computer to external elements'. Programs in this area, as a technology field developed recently using computers, perform certain specific functions for human purposes through connectivity to external input elements or peripherals. Grade 4 corresponds to this. The interface of related programs includes voice, character, image, pattern recognition for human interface or multimedia and moving picture expert group (MPEG), etc. The term interface here is used as meaning the connecting of external elements to a computer system. The third area is 'computer to computer,' which is network or communication of grade 6 to 7.

Table 10. Patent distribution by hardware relativity grades.

| No. | Grade | Related programs | Cases | Rate |
|---|---|---|---|---|
| 1 | 2 | O/S and control relatives | 4,882 | 0.2261 |
| 2 | 3 | Utility | 2,990 | 0.1385 |
| 3 | 4 | Interface | 3,058 | 0.1416 |
| 4 | 5 | GUI, Expert, DBMS, Graphic | 2,576 | 0.1193 |
| 5 | 6 | Network | 2,347 | 0.1087 |
| 6 | 7 | Communication | 3,093 | 0.1433 |
| 7 | 8 | Language/ Compiler | 937 | 0.0434 |
| 8 | 9 | Wordprocessor | 197 | 0.0091 |
| 9 | 10 | Game | 515 | 0.0239 |
| 10 | 11 | Simulation | 400 | 0.0185 |
| 11 | 12 | Spreadsheet | 112 | 0.0052 |
| 12 | 13 | Calculation | 431 | 0.0200 |
| 13 | 14 | Statistics | 52 | 0.0024 |
| Total | - | - | 21,590 | 1.0000 |

### C. Statistic Application

In fact, what should be considered here is that, as practical matter, there are four variables to consider in determining the probability of patentability for computer programs. They are: (a) the development

trend of industrial technology fields, (b) real number of patent applications, (c) patent examination guidelines, and (d) software nature, that is, relativity with hardware.

If development of a certain technology field is flourishing, the number of applications in that field will increase. And the number of patented computer programs related to it will follow from the viewpoint of probability. In addition, the wider the acceptance range for software patent ability is, so also will the number of patented computer programs increase for programs which were hard to patent in the past because examination criterion was tight. software nature also has an influence on the probability of patentability as it is on the degree of difficulty in expressing relativity of computer program with hardware. The Software Nature which is relativity with hardware is an absolute state, but the other three variables are relative and variable.

The results are shown in Table 2 after searching directly the WPI[6] database through the year of 1992 and keywording the related program descriptions for each grade. 'Cases' is the number of patented programs and 'rate' is the occurrence rate with respect to total cases. As grade 1 is almost hardware level and grade 15 is practically not a program, they were excluded from Table 2. Programs of grade 2 which control hardware directly represent the largest case. Communication programs also have many cases notwithstanding being grade 7 because the development of that industrial field is very prosperous.

Those patented in calculation programs here are not for pure calculation method but for its application to apparatus use. Graphing Table 2, it is as shown in Fig. 5.

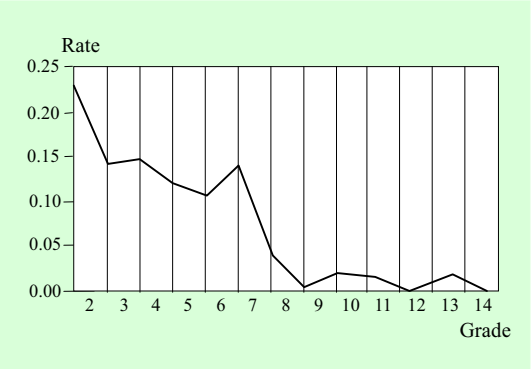


**Fig. 46.** Patent distribution graph.

Approximating this to an exponential function having the intercept of y axis at 1, Fig. 6 is made. The value of d in (1) selects the best-fitted curve. The grade corresponds to degree of software nature and the rate means the patentability. The patentability function demonstrated by the above trend graph is :

$$P = e^{-0.44S}. \tag{2}$$

It can be seen that this also is an average patentability equation since 1992 because the above analyzed data pertain through that year.

To calculate a practial value of $P$, the value of each grade devided by 15, the number of total grades is applied for values of

[6] 'World Patent Index' presented by Derwent

Table 11. Absolute patentability of computer programs.

| G | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | $\infty$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| S | 0 | 0.07 | 0.13 | 0.20 | 0.27 | 0.33 | 0.40 | 0.47 | 0.53 | 0.60 | 0.67 | 0.73 | 0.80 | 0.87 | 0.93 | 1 | - |
| P | 1 | 0.94 | 0.88 | 0.82 | 0.77 | 0.72 | 0.67 | 0.63 | 0.59 | 0.55 | 0.51 | 0.48 | 0.45 | 0.42 | 0.39 | 0.37 | 0 |

∗ G is grade, S is real applied value for software nature, and P is patentability.

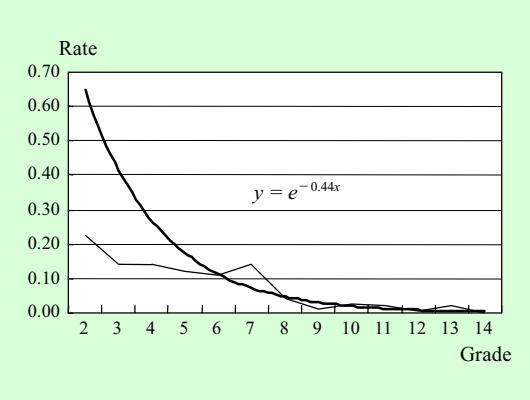


Fig. 47. Approximated exponential graph.

$S$ to match the rate in the range of 1, the intercept value of $P$. Doing so, the function formula is not changed and appropriate values of $P$ to $S$ are taken according to the exponential function.

In (1), $d$ includes three variables, development trend of industrial technology fields, real number of patent applications, and patent examination guidelines, of four variables presented above determining the trend of patentability as a damping coefficient of an exponential function. That is, the wider the acceptance width of examination guidelines, the more prosperous the development trend and the larger the application numbers, the value of $d$ becomes less, inversely proportional, and degree of patentability for computer programs approaching 1 will become higher.

When the value of $d$ is 1, $P$ can be taken as an absolute or standard patentability degree because it is changed by only the value of $S$. The change of patentability at that time by software nature, hardware relativity, is as shown in Table 3.

## 5. Patentability according to Software Nature

As the class of a computer program becomes close to the hardware level, usability of natural law becomes clear, and thus patentability increases. Because operating system controls hardware directly or has a close relationship with hardware, patentability is high.

It is important to note that patentability judgement gets to be difficult in case that the distance from hardware is great as with application programs.

As for application programs of special purposes related to a hardware system such

as communication of middle and large computer, etc., patentable probability becomes high. However, as for application programs for pure user in PC domain, it gets to be difficult to show patentability, and they are decided case by case.

Whether patentable or not is an alternative character from statutory standpoint. However, the degree of patentability for computer programs between patentable and unpatentable may be referenced as an intermediate theory for deciding, planning and framing a patent in accordance with characteristics of computer programs.

## IV. PATENTIZATION TECHNIQUE FOR COMPUTER PROGRAMS

### 1. Technique in a Patent Document

In drafting a computer program patent, this furnishes a key on how to use substantiality of the computer program so as to enable a person having only general knowledge to understand it. In this also, the degree of difficulty varies in proportion to the degree of patentability which is based on the types of computer programs in the same manner. That is, since patentability is high, the degree of difficulty in drafting a patent specification is reduced. And in case of high-class application programs where patentability is low, the degree of difficulty of framing a specification gets to be high. It is important how we will extract, express and embody the patentability which is latent in all computer programs. Connection with devices should be generally considered to explain a computer program in writing down a specification. It is basic to show with a flow chart the procedure executed by the software and the procedure that the part composing the application apparatus of a microcomputer executes. It is good to use, utilizing drawings well, a chart of system configuration to which a program is applied and a block diagram of its main functions which shows the systematic connective relationship among each function of the program [11].

Sound specification and claim drafting is the key to obtaining statutory patents. The lessons to be drawn from the above discussion may not be reduced to simple rules for application preparation. The application by the PTO[7] to applications under examination will certainly complicate the task of the patent drafter. Equivalency issues raised in 101 determinations will not be easily resolved or ease the task of application preparation. Even so, good preparation will facilitate the prosecution of patent applications within the PTO and will, in many cases, overcome 101 rejections.

Though it would be risky to state unequivocally that careful drafting can ensure that all inventions will be held statutory, it can be said that careful preparation of the claims and specification can help most

[7]Patent and Trademark Office

inventions. Recognizing that there is little that can be done to save a poorly-drafted application after it has been issued by the PTO, the time to act is when the patent application is being initially drafted and prosecuted before the PTO. Further, recognize that the nature of the invention and its preferred embodiment dictates how the specification and the claims should be prepared to ensure a statutory patent.

A problem that many patent practitioners have is that they have been taught to prepare patent applications to maximize protection or, in other words, to draft the claims as broadly as possible. As the scope of a claim is broadened, the chances that it will be held nonstatutory increases. As limitations of the specific process or apparatus are stripped away, the chance that the claim will be held nonstatutory increases. The Alappat application, as originally filed, included claims which positively recited a display screen and memory means for storing on intensity data array. Though claim 15 of Alappat without these recitations was held statutory, it required an appeal to the Federal Circuit.

In the final analysis, the breadth of the claim needs to correspond to the present or predicted embodiment to be commercially exploited. If the commercial embodiment does not require the inclusion of a display, for example, then the exclusion of such a recitation merely to achieve a claim of maximum breadth may well result in a 101 rejection. Whether to define specifically in the specification the equivalents of the preferred embodiment is a difficult question. The potential downside of such statements is that the claims may be unduly restricted if other uncontemplated equivalents are readily available. In an application where the preferred embodiment is a computer program, the inclusion of alternative embodiments in the form of circuits may be used to demonstrate the physical nature of the corresponding means for recitations.

Even if the invention consists only of a computer program executed upon a standard computer with a minimum of peripheral devices, it is important that you illustrate the architecture of the computer and its peripherals in a functional block diagram. In particular, show the source of the signals to be processed by the executed program and how the signals are derived and connected to the computer. The program may configure that computer in a unique way, e.g., its memory will be subdivided into files in accordance with the signals to be processed. Further, provide signal or wave diagrams illustrating the "physical nature" of the signals that the program processes [12].

## 2. Strategy for Acquisition of Patent Rights

As for the development trend in the field of computer techniques, innovative development of software has been made on the basis of development of hardware resources. As the hardware is substance,

spatial, time, and technical limit of development follows, the development speed of hardware has risen suddenly at early period and then changed slowly at a certain level. However, as for invisible formless software, spatial limit does not follow, and if only for a novel idea, it flourishes infinitely. And spread is easy with storage media, so productivity and added value of investment are high.

While software development is active like this, the property right to ideas of creators who developed computer programs, that is, an issue about the protection of an intellectual property right became an important matter of concern. By the way, one basic intellectual property right to computer programs includes copyright. So they have a right at the same time of creation. However, as copyright protects expression, a technical idea included in a program fails to be protected. Thus, in the domain of rights protection, limits follow. Therefore patent protection of computer programs is necessary. In addition to this, protection of computer programs came to be generated internationally in the laws of many countries.

In Diamond v. Diehr, 450 U.S. 175 (1981), the United States Supreme Court opened the door to patent protection for computer software by holding that a process for curing synthetic rubber using a mathematical formula in a programmed digital computer was proper subject matter for a patent.

Patentability issues relating to computer software is a complex subject. Generally, software patent applications should be limited to inventions of significant commercial value. If the commercial life time of the software is only a few years, patent protection may not make much sense because it will take at least two or three years for the software patent to issue. Further, the issuance of a patent will destroy the trade secret status of the software. Once a patent issues, the patented features of the invention will be in the public domain.

However, under certain circumstances, software patents are clearly the best way to protect its property rights. For example, in 1994, Stac Electronics won a $120 million verdict when a jury found that Microsoft Corporation had infringed Stac's data compression software patents in the sale of 28 million copies of the MS-DOS Version 6.0 disc operating software. Consequently, Microsoft agreed to pay $43 million and to buy $40 million in Stac stock to settle the dispute [13].

As for the strategy to protect an intellectual property right for a computer program effectively, where the technical character is strong, it is protected with a patent. For programs when patentability degree is low because of strong abstract applicability and the life cycle is short depend on copyright protection.

## V. CONCLUSIONS

Especially to discuss the patentabiliy of

computer program is for establishing a theory as to what the patentability of computer program is. It was made under the historical background that adverse criticism about patentability of computer programs has been since the presentation of protection with a patent right for computer programs first appeared, because of their feature originated from human logical thought. In this situation, there is room for discussion based on many judicial precedents and countermeasures about future system establishments, this paper tried to disclose the essence of patentability of a computer program based on its relation to hardware.

The computer program exists for a computer. And, without a computer program, a computer has no meaning of existence. This means that a computer becomes one complete existence when hardware and software combine like a man composed of a body and a soul to move and rule the body. A computer program loaded in a computer, which is useful for human life, has basically patentable character, that is, potential patentability under the premise that a patent should have substantiality, which is usability of natural law or subject matter. It can be said that such patentability is not uniform but that it differs in degree in accordance with every kind and type of computer program, and that expression technique in writing a specification is needed for acquisition of property right.

Computer programs have class nature in relativity to hardware. There are programs which have close relationship to hardware or are oriented to users. The hardware relativity is strong in the programs which control hardware directly. Applicability for user convenience is strong in programs which simply operate on a computer and merely use hardware rather than control it. In other words, software nature is strong in this case.

Computer programs can be classified into 15 grades according to hardware relativities from strong to low. The zero level of software nature is hardware itself. It becomes software itself when the grade is infinity. The patentability rate of hardware itself, '0' grade of software nature, is 1 or 100%. And the patentability of software itself having no hardware relativity approaches to 0. Therefore we can see that patentability of a computer program basically exhibits an exponential type function which has a value of 1 at the zero exponent value and a negative exponent part. Software naure, $S$, is an independent variable here.

Basic patentability degrees for computer programs are determined when the function value is 1 because the patentability, $P$, is varied purely by the variable $S$.

The coefficient of the exponent part can be viewed as a damping factor, and includes three variables which are development trend, application numbers, and examination guidelines. The damping coefficient is inversely proportional to three variables and probability of patentability increases as the three variables become larger.

This result from engineering analysis could be utilized for decisions in developing a strategy for patenting of general computer programs.

It is not difficult to take out a patent for operating system programs, for they control hardware. But application programs vary exponentially in patentability according to distance from a hardware character and patentability is determined by their types. Potential patentability can be extracted from their applicability. Also in a patent specification, descriptions of software as far as possible in terms of hardware is needed to show substantiality and to optimize claim drafting.

## REFERENCES

[1] Gervaise Davis III, Esq., *Software Protection*, Van Nostrand Reinhold Company, 1985, pp. 142-161.

[2] Howard T. Markey, "Patentability of Mathematical Alogorithms in the United States," *Int'l Review of Industrial Property and Copyright Law*, Vol. 22 (Special Issue), 1991, pp. 986-993.

[3] Andrew R. Basile, "Software-Related Inventions Held Patentable," *Intellectual Property Litigation*, American Bar Association, Sep. 1994, p. 1.

[4] Yosihuji Gousalckw, *An Introduction to Patent Law*, Daekwang, Seoul, 1990, pp. 120-121

[5] J.F. Villella, Jr., "Recent Trend of America about the Patentability of Computer S/W," *'96 Patent Seminar of Computer S/W-Protection Trend of America, Japan and Korea*, Korea Industrial Property Office, June. 1996, p. 39,41,43,85.

[6] Roger E. Schechter, *Selected Intellectual Property And Unfair Competition-Statutes, Regulations And Treaties*, West Publishing, 1993, p. 366.

[7] Cary H. Sherman, Hamish R. Sandison, and Marc D. Guren, *Computer Software Protection Law*, the Bureau of National Affairs, 1991, pp. (101-15)-(101-16), pp. 403-41.

[8] *The Encyclopedia of Mathematics(I of II)*, Korea Dictionary Research Publishing, Seoul, 1995, p. 738.

[9] *English-Korean Dictionary Of Computer*, Kyohak Publishing, Seoul, 1994, p. 996, pp. 1196-1197.

[10] E. Robert Yoches and Esther H. Lim, "The Current State of Patent Protection for Software-Related Inventions in the United States," AIPPI-Korea 1997 International Seminar, May 29, 1997, p. 169.

[11] S/W Study Team of Patent Research Group in West Japan, *The Software, Protection Strategy as a Patent Right*, Korea Invention Promotion Association, 1995, p. 73.

[12] R. Lewis Gable and A. Brian Dengler, "What Can Practitioners Do in Light of Alappat, Lowry, Trovato and Similar Cases?," http://www.juris-diction.com/gabl0001.htm, May 24, 1997.

[13] R. Mark Halligan, Esq., "How To Protect Intellectual Property Rights In Computer Software," http://www.execpc.com/~mhallign/computer.html, May 28, 1997.

**Sang Mu Lee** received the B.E. degree in electronics engineering from Dankook Univerisity, Seoul, Korea in 1989. He worked in the Motorola Korea Ltd. from 1988 to 1990 as a processor engineer. He has worked for ETRI since January 1991 taking part in projects related to integrated management of intellectual property. He has been especially interested in the patentability of computer software.