International Lawyer

Volume 33 | Number 1 Article 6

1999

Examination of the Interrelationship Among the Software Industry Structure, Keiretsu, and Japanese Intellectual Property Protection for Software

Reiko Mashima

RIEKO MASHIMA*

Examination of the Interrelationship Among the Software Industry Structure, *Keiretsu*, and Japanese Intellectual Property Protection for Software

Prepackaged software sales in Japan presently account for approximately fifteen percent of software industry revenues;[1] its U.S. counterpart receives seventy percent of industry revenues.[2] In Japan, customized software has dominated the software market.[3] Given the fact that much of the current best-selling prepackaged software in Japan is of U.S. origin,[4] the underdevelopment of the Japanese prepackaged segment becomes readily apparent. In the dominant custom segment, the comparatively larger software firms are *keiretsu*[5] companies, while most companies in the prepackaged software segment do not belong to *keiretsu*. The

*Rieko Mashima has an LL.B. from Tokyo University, an LL.M. from Harvard Law School, and is a Member of the New York Bar. Ms. Mashima is currently an attorney with Matsuo & Kosugi in Tokyo, Japan and can be contacted at mashima@mknet.chuo.tokyo.jp. The author thanks Professor Robert P. Merges of Boalt Hall School of Law at U.C. Berkeley, Professor Daniel H. Foote of the University of Washington School of Law, Frank L. Nelson, Jr., Esq. of Matsuo & Kosugi, and Dr. Richard L. Thurston of Haynes and Boone, Dallas, Texas.

1. Japan Information Service Industry Association [hereinafter JISA], Jyoho Service Sangyo Hakusho 40 (1994) [hereinafter The White Paper 1994]. Because the Ministry of International Trade and Industry (MITI) supervised The White Paper 1994, it is a reliable resource regarding the Japanese software industry; consequently, I cite heavily from this source. The fifteen percent ratio (of prepackaged software sales) has remained the same in the 1998 White Paper as well.

2. Robert P. Merges, *A Comparative Look at Property Rights and the Software Industry, in* THE INTERNATIONAL COMPUTER SOFTWARE INDUSTRY: A COMPARATIVE STUDY OF INDUSTRY EVOLUTION AND STRUCTURE 273 (David Mowery ed., Oxford Univ. Press 1996) [hereinafter Merges, SOFTWARE INDUSTRY].

3. The White Paper, *supra* note 1, at 40. The software industry can be divided into two main market segments: prepackaged software, sold "off the shelf" as a commodity product, and "custom programming services," where a comparatively large software product is created for a particular client. Merges, SOFTWARE INDUSTRY, *supra* note 2, at 273.

4. *Software Uriagedaka Ranking*, NIKKEI PASOCON (Mar. 1992 & Mar. 1995).

5. Simply speaking, *keiretsu* is an industry group connected by reciprocal shareholdings.

keiretsu structure is now in the midst of change due to the country's economic distress, and Japanese companies are paying more attention to current profits than to *keiretsu* ties.

In his new paper, Professor Robert Merges attributes the discrepancy in Japanese custom and prepackaged software companies to the interrelationship between comparatively weak intellectual property (IP) protection for software in Japan and Japanese software industry structure.[6] Professor Merges initially notes that legislative IP protection makes prepackaged software business feasible by replacing individual contract negotiations. He also points out that strong IP rights attract investment capital to firms where a copyright law protects software and software companies own the copyright of software; investors can take the copyright of companies' software in case of default. For example, investment companies can sell or lease the software to another company to continue its development. Professor Merges then reasons that because Japanese software firms in the dominant custom segment protect their software by implicit contracts with a few closely-allied, particular customers (within the *keiretsu*) and not by legislative IP rights protection, that Japan kept its legislative IP software protection weak to allow Japanese software companies to appropriate foreign ideas at lower cost. Thus, Professor Merges concludes that weak legislative IP protection has caused the underdeveloped prepackaged software segment in Japan.[7] Also, Merges states that a *keiretsu* software company is locked in each *keiretsu* group partly because

6. Merges, SOFTWARE INDUSTRY, *supra* note 2, at 275, 282, 286-87, & 290.

7. In the author's view, the business reasons accounting for the small market share of prepackaged software in Japan are the attitude of Japanese users, the hardware environment, and the difficulty software companies experience obtaining financing. The mainframe market was so fragmented that it gave insufficient incentives for developing prepackaged programs to software companies. Also, Japanese customers with abundant funds preferred custom software both to satisfy their detailed business needs and to differentiate themselves from competitors.

In the Japanese personal computer (PC) market, however, a dominant platform existed from the mid-1980s, the NEC PC-98 (not IBM compatible), holding sixty percent of the market share. The PC-98 drew a sizable number of prepackaged programs.

The most likely explanation for the small prepackaged software market in Japan is the limited size of the NEC PC-98 market when compared with the IBM PC platform. On the demand side, the ratio of PCs to mainframes was far smaller in Japan than in the United States, though prepackaged software has been used for PCs even in Japan. Moreover, specialized word-processing machines, *wahpros*, were popular and exceeded PCs in shipment until recently. On the supply side, even the PC-98 platform was still too small to induce a shift to an investment-first, competitive prepackaged software business from the lucrative and sizable custom business for mainframes. Custom software was constantly in short supply until the early 1990s, and the market was four to seven times as large as the market for prepackaged software. Development of custom software was paid either by the hour regardless of the quality and efficiency of the development work or by a lump sum payment upon attainment of certain milestones. Customers guaranteed receipt of the developed software and software companies bear only operation costs until the last payment by a customer. On the other hand, in the prepackaged business, companies must not only bear development costs but also deal with the uncertainty of whether they can obtain adequate revenues through sales and licensing upon completion.

To make "economies of scale to R&D" and timely reinvestment happen in an R&D-driven high-technology industry, the existence of a sizable market is crucial. Further, "J-curve profits" due to

if it left the *keiretsu* it would lose work from the other *keiretsu* group companies, and partly because it would not be able to easily obtain financing.[8]

This article will: (1) show that some assumptions of Professor Merges' formulation are incorrect; (2) convey a more realistic picture of the Japanese software industry; and (3) explore the impact of the changing *keiretsu* structure on the industry. Part I of this article examines Merges' *keiretsu* hypotheses; it also explores the ongoing change of the *keiretsu* structure and its effects on the software industry, taking into consideration the tough financing situation for Japanese software companies. Part II analyzes the interrelationship between the IP rights regime in Japan and its software industry structure. Finally, Part III explains Japanese IP software rights protection since the 1980s and analyzes whether it has been weaker than that in the United States.

I. The Impact of the *Keiretsu* Structure on the Japanese Software Industry

Most of the larger Japanese software companies are *keiretsu* companies whose main business is the development of custom software. The basic attributes of the *keiretsu* system include cross-holdings of shares by the companies accompanied by implicit mutual agreements not to sell such reciprocally-held shares. Situated at the center of a *keiretsu* is a main bank (a lead lender to the group), a large industrial firm, and/or a large trading company.[9]

Section A describes the development of *keiretsu* software companies and the traditional structure of *keiretsu*. Section B examines Professor Merges' hypotheses that: (1) a *keiretsu* software company is locked in each *keiretsu*; and that (2) implicit contracts in the dominant custom segment caused weak IP protection in Japan. This section also analyzes the impact of the ongoing change of the *keiretsu* structure on the software companies.

---

minimal production costs of programs boost the attractiveness of a sizable market to software companies. IBM PCs provided 'network externalities,' which accounted for the prosperity of the U.S. software industry—the power of standardization through IBM machines (including its clones) and MS-DOS. Unlike IBM, NEC did not adopt an open architecture policy, remaining hostile to clone manufacturers. Moreover, while it sold IBM clones outside Japan, NEC did not try to bridge the incompatibility with the IBM PC. The Japanese prepackaged software business was limited to the NEC PC-98 platform: one-fourteenth of the worldwide IBM platform. Consequently, the NEC PC-98 platform gave far fewer incentives to Japanese software companies than the IBM PC platform did to U.S. companies, especially when profitable and sizable custom business was abundant. For more details, see Rieko Mashima, *The Turning Point of Japanese Software Companies*, 11 BERKELEY TECH. L.J. 429 (1996) [hereinafter Mashima, *Japanese Software Companies*].

8. Interview with Robert P. Merges, Professor of Law, Boalt Hall School of Law, University of Califonia at Berkeley, in Berkeley, Cal. (July 20, 1995).

9. Only one-tenth of one percent of Japanese corporations belong to a *keiretsu*. However, member firms account for approximately one-quarter of total corporate sales and paid-up capital of all Japanese corporations, and represent one-half of all listed Japanese corporations. W. CARL KESTER, JAPANESE TAKEOVERS: THE GLOBAL CONTEST FOR CORPORATE CONTROL 55 (1991).

A. *KEIRETSU* SOFTWARE COMPANIES: ORIGIN AND DEVELOPMENT

During the 1960s, large banks and securities companies hired software engineers to develop their information systems (*daiichji on-line*). On average, a system department of a large commercial bank had approximately 200 to 300 employees, including system operators. After accomplishing the *daiichji on-line*, the banks and securities companies changed their systems departments to subsidiaries to allow them to obtain work from the outside. As the Japanese Antimonopoly Law prohibits banks from receiving work from other companies, banks' spinning off their information system departments as independent entities was inevitable to pursue such outside business.

From the latter half of the 1980s until the early 1990s, when people were afraid to predict a 'software crisis' (a serious shortage of software engineers and software supply), hardware manufacturers and CSK, a large software company, established many local software companies, partly due to a cheaper labor force and partly because a number of software engineers wanted to work close to their hometowns to avoid poor housing conditions and other problems associated with large cities. While several software companies, sometimes as many as fifty to seventy, usually exist within one *keiretsu* group, from this author's experience, they do not work jointly.[10]

B. EXAMINATION OF MERGES' *KEIRETSU* HYPOTHESES AND THE CHANGING *KEIRETSU* STRUCTURE

Some of Professor Merges' assumptions (such as trade partners and contractual practice) differ from the software industry practice in Japan and tend to overestimate the effect of *keiretsu*. In this author's experience as in-house counsel for a *keiretsu* software company—one of the largest and oldest software companies in Japan—and from interviews conducted with industry executives and trade association personnel, Professor Merges' assumptions should be revisited. In the course of examining Merges' hypotheses, the reality of *keiretsu* transactions in the software industry setting is also introduced here.

Professor Merges' hypothesis is based on the following perception of *keiretsu* software companies, formulated, in part, by relying on some presentations at the International Computer Software Industry Project at Berkeley in 1992:

> [M]ost software companies in Japan are affiliated at least informally with large Japanese companies or groups of companies—the famous *keiretsu* structure. Reports in the early 1990s show that in Japan, the majority of software is developed in-house or is commissioned from independent (but affiliated) companies. Experts on the industry state that even today, each software firm normally supplies only one company.[11]

10. *See also* Telephone Interviews with Akira Uchinuno (May 1995).
11. Merges, SOFTWARE INDUSTRY, *supra* note 2, at 281.

While fairly large Japanese software companies in the custom service segment are *keiretsu* software companies, many smaller software companies are not.[12] Also, in the prepackaged software segment, most companies do not belong to *keiretsu*.[13] In number, mid-sized or small-sized software companies constitute the majority of the 7,000 software companies in Japan.[14]

More importantly, *keiretsu* software companies are not limited to a single trade partner.[15] In rare cases, a few software companies that are a part of a hardware manufacturer *keiretsu* and that deal with operating systems might supply only one company.[16] Trade partners of most *keiretsu* software companies, however, include several, even dozens of companies both inside and outside of the *keiretsu* group.[17] In sum, Merges' assumptions overestimate the effect of *keiretsu*.

1. *Merges'* Keiretsu *Hypotheses (i): A* Keiretsu *Software Company is Locked in its* Keiretsu

Professor Merges states that a *keiretsu* software company is locked in each *keiretsu* group because if a software company should attempt to leave its *keiretsu*, not only would it lose work from the other companies within the same *keiretsu* group, but it would also not be able to obtain financing easily.[18] This theory is generally correct although, as discussed above, a *keiretsu* software company does have trade partners both inside and outside of the *keiretsu* group,[19] and this trade partner diversification will continue as Japanese companies strengthen their focus on cost performance. Also, a primary advantage for a *keiretsu* software company is the ease of obtaining financing.

a. Changing *Keiretsu* Structure's Influence in the Software Business Context

Due to the necessity of restructuring and/or to enable a company to report profits during the current severe recession, the Japanese *keiretsu* structure is in the midst of change. Not only are *keiretsu* companies selling their cross-held

12. Yukio Ohno, General Manager of the Legal Affairs Department of The Japan Research Institute, Ltd., guesses that there are approximately 500 *keiretsu* software companies. Though guessing, Ohno believes that the majority of software companies in number are not *keiretsu* software companies. Interview with Yukio Ohno (June 2, 1994); *see also* JISA Membership Directory.

13. Judging from their company names and major shareholders, most members of JPSA (a trade association of PC software companies) do not appear to be *keiretsu* companies. *See* JPSA Membership Directory, 1995.

14. MITI Survey (1992). More than half of the 17,000 companies maintain a small office of less than thirty employees. The top 100 companies generate thirty-six percent of the industry's sales revenues. The White Paper 1994, *supra* note 1, at 71.

15. Telephone Interviews with Akira Uchinuno, Legal Department Manager of Hitachi Information Systems (Mar. 1, 1996) & Akira Tanaka, former Chief Engineer of NEC (Mar. 10, 1996).

16. Telephone Interviews with Akira Tanaka, *supra* note 15 & K. Shiotsuki, Fujitsu Public Relations, N.Y. (Mar. 5, 1996).

17. Telephone Interviews with Akira Uchinuno, *supra* note 15 & Akira Tanaka, *supra* note 15.

18. Interview with Robert P. Merges, *supra* note 8.

19. *See supra* introduction to Part B.

shares,[20] but corporations have been paying more attention to cost performance, which inevitably results in less emphasis on the *keiretsu* relationship.[21]

Selling cross-held shares is aimed at bringing in large amounts of capital, because equity investment in a Japanese company commands a substantially greater asset position than the stock price may suggest. This fact is due to hidden assets, primarily consisting of equity investments for cross-held shares and land. They are both carried on the balance sheet at their purchase prices, even after their market values have appreciated many times over.[22] The trend of selling cross-held shares will continue, because Japanese accounting practice will begin to apply a market price method to financial products for the settlement of accounts in fiscal year 2000, and to cross-held shares in fiscal year 2001, thus eliminating the magic of "hidden assets."[23] Also, now that the Japanese economy's high growth period is over, the basic assumption that the trade volume will increase in the long-term has been shaken, and high costs in the name of *keiretsu* have become an arduous burden.[24]

b. More Competition Regardless of *Keiretsu*

It is basically true that *keiretsu* software companies receive large amounts of work from companies within the same *keiretsu* group. Other than at the time of incorporation, however, a *keiretsu* structure has not always brought sufficient business to a *keiretsu* software company. For example, Canon Software generates sixty percent of its sales revenue from software development—half of which is development for companies outside the *keiretsu* group.[25] The loosening *keiretsu*, with more emphasis on cost performance, has helped bring a more competitive business environment to the software industry, both inside and outside the *keiretsu*.[26]

---

20. Even cross-shareholdings with banks are also declining while banks decrease their equity positions in customer companies to obtain financing to write off bad debts. NIKKEI, July 30, 1995.

21. For example, the Japan Auto Manufacturers Association's announcement of setting its industry standard for EDI (electronic data interchange), necessary to increase efficiency, within 1995 showed an increase of inter-*keiretsu* transactions. NIKKEI, Apr. 21, 1995, at 11.

22. For example, 1986 land price data showed that land owned by First Section Companies in the Tokyo Stock Exchange was 25.2 times as valuable as its book value. KESTER, *supra* note 9, at 160. Also, now that stock prices and land prices have dropped, the merit of cross-holding (hidden assets) has decreased. In addition, crossly-held shares with low dividends block effective use of assets. NIKKEI, Aug. 17, 1995, at 4.

23. NIKKEI BUS., Jan. 11, 1999, at 26.

24. NIKKEI, Aug. 17, 1995, at 4.

25. NIKKEI, May 8, 1994.

26. In addition to the loosening *keiretsu* ties, the ever-increasing use of prepackaged software under hardware downsizing (especially for their core business systems) will increase competition, regardless of *keiretsu* ties. When user companies can afford customized software, they prefer a *keiretsu* software house because they do not want to disclose their business scheme to member companies of competitor groups for the purpose of software development. When information systems integrate more prepackaged software, evaluation of the cost performance and quality of offered plans and services may play a more significant role in user companies' decision-making than whether a software company is within the same *keiretsu* or not. *See* Mashima, *Japanese Software Companies*, *supra* note 7, at 433.

There are two instances when a non-*keiretsu* company can cut into *keiretsu* for work or when a *keiretsu* company can obtain work from another *keiretsu*: (1) when new technological expertise is not offered within a *keiretsu* group; and (2) where the work is offered based on competitive prices and quality of work. Interviews with industry managers confirm this point.[27] For example, Mr. Akira Uchinuno of Hitachi Information Systems explains:

> If *prices and quality of work* are competitive, a non-*keiretsu* [including a company belonging to a different *keiretsu*] can get work within the *keiretsu* group. Before, a non-*keiretsu* company could get work only when such service was not available within the *keiretsu* group. But things are changing partly because of the recession and partly because of pressure from abroad.[28]

Thus, trade partners of a *keiretsu* software company have diversified to include relationships with companies outside of the *keiretsu* group. Indeed, in 1995, out of forty-three *keiretsu* software companies with sales revenues of more than five billion yen, twelve companies earned more than sixty percent of their revenues, and another fourteen companies earned thirty to sixty percent of their revenues from companies other than their parent companies.[29] The *Nikkei Computer Magazine* describes these companies as "independent" and "close to independent," respectively.

c. *Keiretsu* Software Companies Will Still Remain Within the *Keiretsu* Due to Creditworthiness as *Keiretsu* Companies

Keiretsu software companies will remain within the *keiretsu* structure due to creditworthiness based on their *keiretsu* status. Two well-known legal executives in the software industry confirm that obtaining financing is eased due to *keiretsu* company status and the socially-implicit contract that a *keiretsu*'s parent company will not allow its subsidiary to go bankrupt.[30] At the same time, monitoring by a main bank that is backed by its lending power continues, suppressing the opportunistic behaviors of *keiretsu* software companies. This bank monitoring partially contributes to the creditworthiness of *keiretsu* software companies.

A main bank's strong influence over client corporations is based upon its lending power as the largest provider of capital to the *keiretsu* group. However, companies that could obtain funds at a lower cost in the Euromarket used such funds to repay bank loans and became less dependent on banks. Since the 1980s, the main bank's monitoring function within the *keiretsu* has further decreased

27. Interview with Mr. X (Mar. 1995) (Mr. X is an experienced business and marketing executive for a fairly large software company in Japan. He shared his views with the author on the condition that he remain anonymous.).

28. Interview with Akira Uchinuno (Aug. 1994).

29. NIKKEI, Mar. 30, 1998, at 150-51.

30. Telephone Interviews with Akira Uchinuno, *supra* note 10 &Yukio Ohno (Aug. 13, 1995).

as a consequence of both the deregulation of the Tokyo financial market[31] and the stronger cash earnings of Japanese companies from high growth.[32]

However, the chain reaction of financial market deregulation and lower dependence on bank lending did not affect software companies as most of them were not listed in the capital market. The *Wall Street Journal* comments, "Until now, only relatively large companies with years of profits behind them have had access to equity financing. That system has left many of Japan's fastest-growing companies starving for cash."[33] Thus, obtaining financing has generally remained a concern for Japanese software companies.

Even though the *keiretsu* ties have been loosening, a software company's alteration from a *keiretsu* company to a non-*keiretsu* company is unheard of. This is due to the fact that *keiretsu* affiliation allows these companies to obtain financing far more easily than non-*keiretsu* software firms. This is a great advantage in Japan where software companies have difficulty obtaining financing because bank lending focuses on real property as collateral, which most software companies do not have. Also, Japanese venture capitalists, unlike those in the United States, do not actively invest in the high-technology industry, and a functioning NASDAQ-type market does not exist.

d. Poorly Developed Venture Capital in Japan

In Japan, investment by venture capitalists in high-technology lags far behind its U.S. counterpart—including early-stage investment and advisory capability for management. Nevertheless, the total amount of investment has come close to that in the United States.[34] This is due to the weakness of Japanese venture capitalists.[35] In contrast, U.S. venture capital invests twenty-seven percent of their funds in the software industry.[36] A 1993 source states that the U.S. figure

31. In 1980, the Amendment of the Foreign Exchange Control Law permitted cross-border capital transactions with only prior notification of the Ministry of Finance. Obtaining a formal permit became unnecessary. KESTER, *supra* note 9, at 188-89.

32. *Id.* at 13-14, 187-97; "The conclusion is that dependence on a bank is no more to the liking of Japanese management than management in other countries, and for leading Japanese companies no longer a significant issue." JAMES C. ABBEGLEN, & GEORGE STARK, JR., KAISHA: THE JAPANESE CORPORATION 189 (1985).

33. Robert Steiner, *Japan Launches Small-firm Stock Market*, WALL ST. J., July 19, 1995, at A5.

34. *See* Mashima, *Japanese Software Companies*, *supra* note 7, at 452-53.

35. NIKKEI, Apr. 17, 1995, at 16.

36. *Id*; *see also* Yasunori Baba et al., *The Japanese Software Industry: The "Hub Structure" Approach*, 24 Research Pol'y 474, 478 (1995):

> The second problem is that compared with the U.S., venture capital scarcely flows into the Japanese software industry. According to a survey in 1989, 11.0% of American venture capital flows into the Japanese software industry, but the ratio in Japan was only 0.04% (Nikkei, Nov. 24, 1992). The reason seems to be that venture capital holders in Japan are not always well-versed in technological assessment and often have difficulties calculating the returns to an investment in an expertise-intensive company like a software house. Therefore, in contrast to the U.S., Japan is still not in a position to promote innovative software businesses.

is twenty-one percent which is still a high figure, even compared to other U.S. high-technology investment in areas such as biotechnology (nine percent) and computer hardware and systems (three percent).[37] Because a software company's profits increase in J-shape (high-return) fashion thanks to minimal production costs once a program is developed, a venture capitalist can afford to take a high risk.[38] Being able to take this high risk requires the capability to assess technology—a job at which Japanese venture capitalists are poor.

Also, Japanese venture capitalists do not actively invest in early-stage companies because it takes an average of thirty years for a Japanese venture company to go public.[39] This prohibits venture capitalists from collecting and cashing-in on their investments in companies at earlier stages.[40] As a result, the ratio of such investment is about half of that in the United States, where it takes six years on average to go public.[41] Major Japanese venture capitalists invest sixty percent of their funds in companies that have aged more than ten years since their incorporation date, and only fifteen percent in companies aged less than five years.[42] In general, start-up companies in Japan face difficulty in obtaining financing.[43]

The above thirty-year "to go public" period stems from the fact that the actual threshold to be listed on the over-the-counter market is much higher than its written rule,[44] due to the special rules of the Securities Companies Association.[45] While the rule requires recurring profits of twenty million yen and net assets of 200 million yen, the actual threshold is said to be recurring profits of 300 million yen (about $3,000,000 at a $1=100 yen exchange rate) and net assets of one billion yen.[46] In contrast, on the NASDAQ, the required profit is more than $750,000 before taxes (but this is not required if net assets exceed twelve million).[47] Even companies still in the red can be listed on NASDAQ.[48] Very recently, a second over-the-counter market with lower requirements was created, but it

37. United States Bureau of the Census, Statistical Abstract of the United States, Venture Economics Investor Services (1993), Table No. 854, Venture Capital Disbursements by Stage and Industry at 550 [hereinafter Bureau of the Census].

38. Hisashi Washiyama, General Manager of JAFCO America N.Y., NIKKEI SHINBUN, Feb. 15, 1996, at 26.

39. NIKKEI, Apr. 17, 1995, at 16.

40. *Id.*

41. Bureau of the Census, *supra* note 37. American venture capital investment in early stage companies is 24% (start-up 7%, seed 7%, and other early-stage 10%). The other investment purposes are expansion (55%), LBO (6%), and bridge loan or public purchases (15%).

42. *21-Seikiheno Venture Business*, NIKKEI, Apr. 17, 1995, at 16 (according to a survey by VEC in 1991 and 1992).

43. Out of average start-up costs of approximately twenty million yen ($200,000) in 1995, only fifteen million yen represented loans from financial institutions while the rest was from relatives, friends, and business partners' support. NIKKEI, Apr. 16, 1994, at 1.

44. NIKKEI, Jan. 4, 1995, at 15.

45. NIKKEI, Feb. 1, 1995.

46. NIKKEI, Jan. 4, 1995, at 15.

47. *Magarikado-no Tentokabu Shijoh*, NIKKEI, Aug. 23, 1994.

48. *Id.*

has not been active. In the United States, "angels" and/or venture capitalists invest in venture businesses, and successful businesses typically go public on NASDAQ in three to five years, thus attracting new investment.[49] Such fast investment cycles do not exist in Japan.[50]

Still, if MITI-organized, industry-cooperative development projects had worked, it would have been helpful to the technological development of the cash-starved prepackaged software companies if R&D results were shared. But, unlike its support of the computer hardware industry, MITI-initiated software projects have mostly failed.[51] The failure of the MITI-initiated projects appears to be due to the fact that the speed of technological development in the software field has been much faster than the bureaucrats' planning ability. Even when some of their projects were aimed at developing "tools commonly available for software development," the tools actually meant aids for the dominant custom software. Due to the nature of custom-made software, those tools (even when produced) were not widely applicable.[52] Furthermore, the fact that using participating companies' huge R&D facilities interchangeably was not necessary in the software field, unlike in the hardware industry, decreased the software companies' incentives to commit themselves to industry-wide projects.

To summarize the considerations applied in the first part of Merges' hypothesis, Japanese companies, regardless of *keiretsu* ties, are paying more attention to current profits and cost performance, and thus software companies are more competitive both inside and outside of their *keiretsu*. Still, *keiretsu* software companies will remain within *keiretsu* groups because they can obtain financing far more easily than can non-*keiretsu* software firms, thanks to their creditworthiness. In using Merges' terminology, lower financing costs (creditworthiness) primarily explains why a *keiretsu* software company is locked to its *keiretsu*.

2. *Merges'* Keiretsu *Hypotheses (ii): Implicit Contracts in the Dominant Custom Segment Caused Weak IP Protection in Japan*

Professor Merges also states that implicit contracts in the dominant custom segment caused weak IP protection in Japan:

---

49. *Id.*

50. *Id.*

51. Interview with Mr. X, *supra* note 27; Karl Ruping, Institute of Intellectual Property, *Patent Protection of Computer Software in Japan and the United States* (Mar. 1998) at 10. For example, "MITI supported the formation of the Information Processing Promotion Association (IPA) in 1970 [sic.] [but it] was relatively under-funded and failed to attract the support of government or industry banks. In 1973, MITI instituted the Software Module Project which, despite generous subsidies, never produced the family of software applications that were expected." *Id.* n.20. In 1975, MITI initiated the Program Seisan Gijutsu Keikaku (Program Production Technology Plan) with a budget of five billion yen for five years to develop common tools applicable to various programming languages. Also, MITI started the SIGMA (Software Industrialized Generator and Maintenance Aids) Project, aiming at the automation and mechanization of software development. The planned budget was 25 billion yen. Both projects produced little. *Id.*

52. *Id.*

> [T]he custom services segment of the Japanese industry dwarfs the prepackaged software segment; they are 85 percent and 15 percent, respectively, of the industry. . . . *Because many Japanese software firms are closely allied with a particular customer/patron* [that is, *keiretsu*], they do not depend on formal legal rights to protect *their software. Protection of intellectual property is handled by the (largely implicit) contract between the firms*. [W]ithout an arm's-length market transaction, and the attendant risks of misappropriation of intellectual property, there is less need for formal legal rights.
>
> This suggests . . . . [t]he dominance of the customized software sector made strong legal protection irrelevant. Moreover, because such protection would have restricted access by Japanese firms to the products of foreign nationals, it would have conflicted with the "technology transfer" policy fostered by the Japanese government. Weak protection allowed Japanese firms to profit from foreign ideas at lower cost. *The implicit contractual protection of intellectual property* was an effective appropriation mechanism for the Japanese software firm.[53]
>
> . . . [T]here appear[s] to be a connection between weak copyright protection in Japan and the delayed emergence of the prepackaged segment of the Japanese software industry.[54]

a. No Implicit Contracts

The author disagrees. Even between *keiretsu* companies, written contracts must be executed—the companies do not depend upon implicit contracts. As Mr. Yukio Ohno, former General Manager of the Legal Affairs Department of the Japan Research Institute, notes, "Until the early 1980s, software companies did not make written contracts when dealing with *keiretsu* companies in many cases. However, in the 1980s, software companies (including mid-size and small ones) started to execute written contracts even when dealing with *keiretsu* companies."[55] Implicit contracts are not the case even with the most closely-tied hierarchical *keiretsu*—such as the automobile industry.[56] An experienced employee of a major automobile manufacturer states, "Purchasing or selling goods or services without any documentation even with *keiretsu* parts companies is unthinkable. If done, our accounting department will be in trouble to deal with taxes."[57]

Instead of a company's mere status as part of the *keiretsu*, what matters in reducing bargaining costs seems to be: (1) experiences and length of transactions in the past; and (2) bargaining power. Five experienced businessmen from large-sized software companies and one from a mid-sized software company mentioned that whether a customer is within a *keiretsu* group does not matter to a great extent in contractual negotiations. In the case of a completely new customer, contractual negotiations tend to become lengthy and more detailed. After continu-

53. Merges, SOFTWARE INDUSTRY, *supra* note 2, at 281-82.

54. *Id.* at 281.

55. Interview with Mr. X, *supra* note 27.

56. Telephone interview with an experienced employee of one of the major auto manufacturers in Japan (Feb. 28, 1996). A basic contract is executed and daily delivery and acceptance of parts are based on purchase orders and statements of delivery.

57. *Id.* Mr. Akira Tanaka, former Chief Engineer of NEC states, "NEC executes a written contract even with a 100% subsidiary, let alone with the other companies. This was the case in the 1980s as well." Telephone Interview with Akira Tanaka, *supra* note 15.

ous transactions without trouble, the time and energy expended on contractual negotiation decreases.[58] Thus, bargaining costs decrease over time.

Also, the bargaining power of a company affects the transaction costs. For example, Mr. X's mid-size non-*keiretsu* company can use its contract form even with much larger corporations in offering its competitive and popular system integration service.[59] Mr. Ohno, whose company is the second-largest VAN company (exceeded only by NTT Data) stated, "The company does not change contracts depending upon whether a customer is a *keiretsu* company or not."[60]

b. Legislative IP Protection Still Counts

Next, returning to the relationship between implicit contracts and legislative IP protection, even when explicit contracts are executed, legislative protection is necessary because Japanese-style contracts generally are much simpler than those in the United States. If the terms and conditions of contracts—especially for continuous relationships such as custom software development and its upgrades—are too detailed, trade partners may be offended and feel that the other party distrusts them. Typically, contracts provide that all disputes are to be resolved through "mutually-amicable negotiations." Therefore, legislative protection of IP rights is still important because parties refer to statutes when their contracts are silent or not explicit in a given matter. Further, the lack of interpretive case law in Japan, due to far less litigation than in the United States, increases the relative importance of the statutes themselves.

Indeed in the mid-1980s, JISA, a trade association of custom software companies that, unlike prepackaged software companies, depends heavily upon individual contractual negotiations with users, supported MITI's legislative draft to protect software.[61] Thus, the dominant custom service segment was interested in formal legal protection, even while it was negotiating and executing written contracts. It is true that IP rights are especially important to the prepackaged software business—they matter less in a custom software business, which depends more upon contractual negotiations.[62] The difference is a matter of degree.

To summarize Part I, the current changing *keiretsu* structure (more profit-oriented) is diversifying the trade partners of *keiretsu* software companies reaching outside of the *keiretsu* groups and exposing them to more competition either inside or outside the *keiretsu*. Still, *keiretsu* software companies will remain within *keiretsu* groups due to the ease of obtaining financing thanks to their creditworthiness as *keiretsu* companies, while other software companies suffer from a poorly-developed venture capital market. Second, it is not accurate that implicit contracts in the dominant custom segment caused weak IP protection in

58. Interview with Mr. X (Aug. 1994), *supra* note 27.
59. *Id.*
60. Interview with Yukio Ohno (May 1994).
61. *See infra* Part III.A.1.
62. *See id.*

Japan. *Keiretsu* software companies, mostly in the dominant custom segment, have not depended upon implicit contracts since the 1980s; thus, *keiretsu* ties have not automatically lowered bargaining costs. Even when explicit contracts are executed, legislation has real significance given the relatively simple contractual practice in Japan and the lack of case law. Indeed, a trade association of custom software companies sought legislation to protect software in the mid-1980s, and Japanese laws have provided a relatively sufficient statutory basis for legal protection of software since the 1980s.

II. Relationship Between the Japanese Software Industry Structure and the IP Rights Regime

This part analyzes the relationship between IP rights protection in Japan and the Japanese software industry structure. Professor Merges notes the interrelationship between weak IP protection in Japan and the small prepackaged software segment of the Japanese software industry expressing that:[63] (1) not only does formal IP protection make prepackaged software business feasible, but also (2) strong IP rights attract investment capital to firms.[64] However, Merges notes as a third proposition that implicit contracts, in *keiretsu*, in the dominant custom segment caused weak IP protection in Japan.[65] Thus, he states that weak legislative IP protection has caused the underdeveloped prepackaged software segment in Japan.

While the author agrees with part one of Merges' theory, she disagrees with the remaining two parts, because some of Merges' assumptions differ from the industry practices in Japan. In this section, the effect of software protection by the Japanese legislation is examined from the standpoint of bargaining[66] and monitoring costs. Then, the impact of IP rights protection on financing is analyzed in the Japanese context.

A. Reduction of Bargaining Costs and Feasibility of Prepackaged Software Business under IP Legislation

Concerning the relationship between IP rights, bargaining costs, and the industry structure,[67] Professor Merges states that legislative IP rights protection for

63. Merges, Software Industry, *supra* note 2, at 290.

64. *Id.* at 283, 287.

65. *Id.* at 281-82.

66. *Id.* at 282-83. Though Professor Merges uses the phrase "transaction costs," he appears to mean bargaining costs, considering the context. In general, "transaction costs" include bargaining costs, monitoring costs, and information costs. The information costs are, among others, expenses necessary to determine the size of the potential markets or value of the technology to a licensee. Robert P. Merges, Patent Law and Policy—Cases and Materials 773-75 (1992) [hereinafter Merges, Patent Law and Policy].

67. Merges, Software Industry, *supra* note 2, at 282-83. Professor Merges' theory differs from a simple "incentive theory." Strong IP protection heightens the return to investment, increasing investment and entry into the production of software; weaker IP protection has the opposite effect. *Id.* at 281-82.

software replaces expensive individual contractual negotiation, thus making prepackaged software business feasible. Merges writes:

> [Property rights] reduce transaction costs; or, more accurately, they make certain transactions feasible that otherwise would not be. This view of intellectual property rights, which might be termed the "off-the-rack-contract" view, is especially important for prepackaged software. Product attributes in this segment are general rather than specific to a particular end user. Similarly, intellectual property rights can be viewed as statutory terms of exchange that provide general transaction attributes. . . . These rights matter less in the custom software segment, where both product and transactional attributes are customized to meet the needs of the individual end user. The legal framework that governs exchanges in this market segment, primarily contract law and its closely related cousin, trade secrets, reflects this fact.
>
> . . . [C]opyright law provides an 'off-the-rack' contract between buyers and sellers of intellectual property, taking the place of individually negotiated contracts.[68]

The author agrees with this part of Merges' theory on the following three grounds. First, the Interim Report of the Copyright Council in 1984, which founded the 1985 Amendment to the Japanese Copyright Law (JCL) to extend copyright protection to software, expressed that the increase of prepackaged software should require legislative protection of computer software. The report stated:

> In consideration of the following changes in the environment, more concrete consideration of the legal protection of software is required. . . . Prepackaged software has increased its shipped amount. With respect to software for personal computers, most of the software . . . is prepackaged. Prepackaged software for personal computers is distributed through various channels such as bookstores, department stores and software retail stores.[69]

Here, the Copyright Council recognized the interrelationship between the growing shipment of prepackaged software and the need to protect software under the Copyright Law. This Report formed the basis for the 1985 Amendment to the JCL. As shown by the chart below, the prepackaged software segment in Japan has steadily increased its sales revenues since 1985, although the increase is partly attributable to the constant increase of personal computer (PC) shipments.[70]

Sales Revenues of Prepackaged Software for Personal Computers in Japan[71]

Year	1985	1986	1987	1988	1989	1990	1991	1992	1993
Growth rate from the previous year (%)	N/A	39	27	41	45	32	19	9	14

68. *Id.* at 282-83.
69. The Interim Report of the Copyright Council (1984).
70. JECC COMPUTER NOTE 22 (JEIDA 1993); JEIDA Report.
71. DENTSU SOHKEN, JYOHO MEDIA HAKUSHO 107 (1995) (based on data from JPSA).

Second, Mr. Tamotsu Satoh, senior executive managing director of Intercom, believes that prepackaged software vendors can rely on legislative protection instead of contractual negotiations:

> The company will fight under the Copyright Law concerning what its shrink-wrap license does not mention. The company sells its prepackaged software through distributors, so it is convenient for the company to adopt the shrink-wrap license. Some users, however, even neglect the provisions of its shrink-wrap license because the validity of a shrink-wrap license has not been established because case law does not exist. Therefore, when such a user infringes Intercom's copyright, the company will directly sue the user under the Copyright Law.[72]

Finally, the fact that an association of prepackaged software companies (JPSA) sought strong enforcement of the Copyright Law and an association of custom software companies (JISA) has worked on its model contract,[73] supports Merges' theory.

The JPSA, founded in 1982, has been working hard to eliminate illegal copies—a critical problem for prepackaged software companies. Prepackaged software companies incur debts at the beginning to develop their products and can recover their investments and reinvest their earnings in subsequent development by selling the products. But, copying software is easy, costs little, and can be repeated without causing deterioration to the original program. Unlike the case of custom-made business software, it is difficult to contractually negotiate and monitor each user; a shrink-wrap license is used for convenience. To cope with this problem, the JPSA established its watch-dog department for software protection in 1985. JPSA has also held seminars to educate users and put 18,000 anti-illegal copying posters in prepackaged software shops, corporations, computer schools, and other locations.[74] Thus, the JPSA's effort has been focused on the strong enforcement of the Copyright Law.

In contrast, the JISA tried to protect its members' rights through contractual negotiations. For more than a decade, the JISA has been studying its own model contract for software development in order to help its members' contractual negotiations with users.[75] The JISA published its Model Contract for Customized System Development in 1986 and partially revised it in 1993 and 1995.[76] Noticing this, the MITI organized the Common Frame Committee[77] to provide common contract terms, development procedure (e.g. system planning and

72. Interview with Tamotshu Satoh, Senior Executive Managing Director of Intercom (Mar. 1995). Intercom has developed and sold the best selling communication software.

73. Interviews with Akira Ogata, JISA Manager (Mar. 1995).

74. Yoxo Shimizu, *Packaged Software Kaihatsu-no Chosakuken Mondai-towa*, KENKYU KAIHATSU MGMT., Dec. 1993, at 71.

75. Interview with Akira Ogata, *supra* note 73.

76. JISA, *Model Contract of System Software Development Comments*, Dec. 1994, preface.

77. Interview with Akira Ogata, *supra* note 73.

installation), and price estimation for custom software development (the Common Frame).[78] Users are gradually accepting it.[79]

In sum, the Japanese Copyright Council's recognition of the necessity of copyright law protection for prepackaged software, Mr. Satoh's statement, JPSA's (prepackaged segment) focus on the enforcement of the Copyright Law, and JISA's (custom segment) contractual approach, all show that legislative IP rights protection makes the prepackaged software business feasible as an off-the-rack-contract and is especially important for prepackaged software.

B. No Effect on Monitoring Costs

Next, the problem of monitoring costs cannot be solved by simply strengthening IP rights unless a compulsory investigation right is included.[80] It is well-known that enforcing a technology license is difficult. First, with respect to copyright, making an illegal copy of a computer program is easy and costs little. Moreover, copying can be repeated without causing deterioration to the original program. Also, with respect to patents, a licensor is mainly concerned that: (1) a licensee may incorporate the patent technology into more products or sell more units than he reports; and (2) the licensee is only entering into the deal to acquire the technology, that is, after selling a few units, he may terminate the agreement and introduce a product incorporating similar, but non-infringing technology.[81]

Though a compulsory investigation right does not currently exist even under the U.S. Copyright Act or the Patent Act, broad discovery in the U.S. litigation system helps investigation and monitoring. Japanese software companies in the prepackaged software business are especially concerned about illegal copying.[82] According to a 1994 report issued by the Software Publishers Association, the rate of illegal copying stands at fifty-six percent.[83] Japanese prepackaged software companies have tried to tackle this problem through user education and criminal accusation.[84]

C. Little Effect of IP Rights on Financing in Japan

Professor Merges comments that legislative protection of software helps software companies obtain financing because: (1) legislative protection reduces trans-

78. JISA, *supra* note 76, at 123.

79. Interview with Akira Ogata, *supra* note 73. Partly to integrate the Common Frame to its model contract and partly to follow new hardware environment (downsizing, open systems, etc.), JISA further updated its model contract in December 1995. JISA, *supra* note 76, preface.

80. A compulsory investigation right authorizes a licensor to investigate business places and books of a licensee.

81. Merges, Patent Law and Policy, *supra* note 66, at 774.

82. *See* Mashima, *Japanese Software Companies*, *supra* note 7, at 445.

83. SPA News Release, Feb. 24, 1995. The piracy rate in the United States was twenty-five percent.

84. *See* Interview with Fuminobu Satoh, JPSA (Mar. 13, 1995).

action costs (bargaining costs), and thus the required amount of funding can be smaller; and (2) when a copyright law protects software, and software companies own the copyright of software, investors can take the copyright of companies' software (e.g., in case of default to sell or lease to another company to continue the development of the software).[85] In Japan, however, the strengthening IP protection did not contribute to the latter because financing systems for venture businesses differ from those in the United States for the three reasons discussed above.[86]

1. *Strong Financial Backing is Crucial, Especially for Prepackaged Software Business*

In the prepackaged software business, companies must bear both development costs and the risk that the market may not accept the new product. In contrast, in the custom software business, it is virtually guaranteed that the developed software will be received by the customer. Software companies bear only operation costs until the last payment by a customer.

Unlike the custom business, the prepackaged software business requires not only up front investment, but also a higher ratio of R&D costs to sales revenues to support innovation.[87] Strong financial backing is therefore crucial to enter and be successful in the prepackaged software business.[88] Further, investment in the prepackaged software business is becoming more costly. The *Nikkei Computer Magazine* estimates an average development cost for the first version of a new prepackaged program to be 100 million yen ($1 million under the exchange rate of $1 = 100 yen), regardless of the applicable hardware type. This figure is based on an average term of development of two years with less than ten software engineers.[89] Mr. Yoshinobu Watanabe, Director of Dynaware, says that currently if a software title does not support at least sales of 100 million yen, software houses will be in red ink.[90] In addition, a software executive states:

> Recent competition has raised marketing costs up to 50 million through 100 million yen. In addition, a company needs to keep the same ten engineers for upgrades, which costs another 100 million yen. Therefore, a software house cannot earn profits if the sales revenues of a new program are less than 300 million yen (3 million dollars).[91]

---

85. *See* Merges, SOFTWARE INDUSTRY, *supra* note 2, at 287 (discussion with Professor Merges (July 20, 1995); "[S]trong intellectual property rights attract investment capital to small firms . . .").

86. *See supra* Part I.B.1(2)(ii). First, bank lending practices focus on real estate collateral. Second and most importantly, venture capitalists are not high-technology oriented and are poor at evaluating the value of software. Third, a NASDAQ-type capital market does not exist.

87. NIKKEI, Nov. 1, 1993, at 136.

88. *See* Mashima, *Japanese Software Companies, supra* note 7, at 452.

89. NIKKEI, Feb. 7, 1994, at 62.

90. Interview with Yoshinobu Watanabe, Director of Dynaware (Mar. 17, 1995).

91. Interview with Mr. X (Mar. 16, 1995), *supra* note 27.

2. *Practical Problems of Software as Collateral—Bank Lending Focused on Real Property as Collateral*

The Japanese Copyright Law assumes that a pledge can be established on copyrights and notes its effect.[92] The Japanese case law also permits the establishment of the right of mortgage on copyrights. Nevertheless, in Japan, IP rights are rarely used to obtain financing. In Japan, almost forty percent of industrial finance is supplied by bank lending, which traditionally requires real property as collateral.[93] But most software companies do not own real property. On average, only seventy cases involving the use of patents and trademarks as collateral occur each year.[94] Most are used as trade credits, while only a few are used for lending by financial institutions.[95] Using software as collateral is much less frequent.[96] Since the 1986 inception of the program registration system, the registration instances to establish the right of pledge and mortgage has remained at less than twenty.[97]

Mr. Tashiro, formally of the Japan Development Bank, points out three problems affecting the effectiveness of software as collateral: (1) vagueness of content; (2) risk; and (3) difficulty in evaluating the value of software as collateral.[98] Vagueness of content refers to updates, the complicated relationship of IP rights of derivative works (customization of prepackaged programs and porting of foreign programs into Japanese), and the prohibition of assignment clauses in license contracts. Risk always accompanies financing, especially for high-tech industries like the software industry. The *Nikkei Computer Magazine* has mentioned that difficulty in evaluating the value of software as collateral, as well as an underdeveloped market to sell software collateral, are reasons for banks' unwillingness to accept software as collateral.[99]

Now even in the United States, banks are not active investors in the software industry—venture capitalists (before a software company goes public) and capital market firms (after going public) are its investors.[100] U.S. banks consider software as supplemental collateral.[101] Software companies in Japan, however, continue to suffer from poorly developed venture capital markets. Strengthened IP protection for software in Japan has not helped software companies obtain financing.

Although Japanese game software companies are not *keiretsu* companies that can easily obtain financing,[102] they have been successful and internationally com-

92. Japanese Copyright Law art. 66.
93. Yasuhisa Tashiro, *Chitekizaisanken-no Tanpo-Toshiteno Katsuyo-nitsuite [Use of Intellectual Property Rights as Collateral], in* RESEARCH OF BASIC PROBLEMS ON THE ECONOMIC EFFECTS OF INTELLECTUAL PROPERTY RIGHTS 281 (Institute of Intellectual Property ed., 1993).
94. *Id.* at 283.
95. *Id.*
96. *Id.*
97. *Id.*
98. Interview with Yasuhisa Tashiro (Jan. 1995).
99. NIKKEI, May 13, 1996, at 91.
100. Tashiro, *supra* note 93, at 291.
101. *Id.* at 290.
102. *See supra* Part I.B.1.b.

petitive.[103] The following conditions (such as a large market)—unique to the game software field in the 1980s—account for their exceptional success. First, the target population (market) for game software includes virtually the entire nation (even though many customers of game software are not adults), and many popular game programs sold millions of copies. In the early 1980s in Japan, PCs were very expensive and only research institutes and a limited number of companies owned them, decreasing the potential market for prepackaged business software.[104] In contrast, a popular Nintendo game machine cost a mere 30,000–40,000 yen ($83–$111 U.S. under the then-current exchange rate of $1=360 yen), including game software. Playing game software was a completely new amusement, and a popular game program could sell millions of copies. In the 1980s, many million-seller programs appeared. Even though the unit price of game software was far cheaper than the prepackaged business software, million-seller game programs rewarded their software developers handsomely.[105]

Second, because the product cycle of game software is much shorter than that of business software, the huge earnings from sales of game software occurred in a short period of time after a game's release. Therefore, a game software company that developed a hit program could quickly recover its initial investment and promptly reinvest its profits in hiring increasingly capable software engineers to develop even more game programs.[106]

Third, because the development cost of a game program was not high in those days, entering the game software business was relatively easy, which further helped this "pioneer era." Thus, the existence of a far larger market for game software (relative to business software) since the early 1980s—together with a fast cycle of investment,profits, and reinvestment in the game software field—resulted in a 'financing-blues free' environment for the game software field, enabling its exceptional success in the underdeveloped Japanese prepackaged software segment.[107]

103. While Japan exported only 13.5 billion yen worth of software (excluding game programs in 1994, it imported 259.5 billion yen worth of such software. However, when the game software was included in the statistics, Japan's import and export of software were almost equal. NIKKEI, Nov. 2, 1995, at 11.

104. Even as of 1996, "[t]he absolute size of the Japanese PC market is approximately one-fourth of that in the U.S. . . . even though the Japanese population is half that of the U.S." Mashima, *Japanese Software Companies, supra* note 7, at 440.

105. The existence of a sizable market is crucial for a high technology industry because a high technology industry requires a high ratio of R&D expenses to total expenses. R&D expenses are fixed charges; therefore, the more units of developed products that are sold, the lower the R&D costs per unit. Thus, a large market allows a software company to spend more on R&D because these costs can be spread over a large number of units. . . . In a huge market, a software company can achieve necessary sales revenues more easily in a shorter period of time than in a small market, even if its individual market share is small. *See id.* at 439.

106. [A] high technology company must earn large profits from a new product quickly, before competitors introduce similar products, so the company can both recover its investment at an early stage and invest in new R&D in a timely manner. Early investment is also important for a company to enable it to generate revenue to reinvest in upgrade developments. *See id.*

107. The game software information is based on an interview with Mr. Kouzuki Kagemasa, President of KONAMI, in December 1996. Mr. Kouzuki adds that nowadays, due to harsh competition, a million-seller game program in the Japanese market is almost impossible.

To summarize Part II, legislative IP rights protection for software in Japan made the prepackaged software business feasible by replacing expensive, individual contractual negotiations, but it did not reduce monitoring costs. More importantly, legislative IP rights protection for software did not help Japanese software companies to obtain financing because: (1) Japanese venture capitalists are not skilled at evaluating the value of software; (2) Japanese venture capitalists are unwilling to invest in early-stage companies due to the lack of a NASDAQ-type capital market; and (3) bank lending focuses on real property. The causal link that Professor Merges assumes exists between IP rights protection and the underdeveloped prepackaged segment of the Japanese software industry structure is disconnected here. The remainder of Professor Merges' hypothesis concerning legislative IP protection will be examined in Part III, and the fact that the software industry did not affect the legal scheme of copyright protection is also explained. For the author's concise analysis of the *Keiretsu* influence on transaction and financing costs, please see the appendix.

III. Intellectual Property Rights Protection for Software in Japan—Is It Weaker than Protection Provided in the United States?

This part examines whether legislative IP protection for software in Japan is as weak as Professor Merges assumes. Until very recently, copyright has generally played a major role in protection for software both in Japan and the United States. Patents on software-related inventions receive the next priority in this chapter, including Japan's new examination guidelines published in 1993 and 1997. Basically, copyright protects the literal expression of software code against copying.[108] "Look and feel" seeks to protect the various nonliteral elements of a computer program such as screen displays and the menu structure. On the other hand, patent law protects "the general inventive concept specified in patent claims, such as a method for performing some software function or operation."[109] Trade secret protection is basically contract-based, and so it is not explored here other than to point out that the Japanese Unfair Competition Prevention Act was revised in 1991 to protect trade secrets and that the Japanese Civil Procedure Law was partly revised in 1996 to protect trade secrets in litigation.[110]

108. A description of a machine protected by copyright only prevents others from copying and does not prevent others from writing a description of their own or from making and using the machine. Unlike patents, independent creation can be a defense to a copyright holder.

109. Merges, SOFTWARE INDUSTRY, *supra* note 2, at 275.

110. *See* Kazuko Matsuo, *Recent Amendment to the Unfair Competition Prevention Law for the Protection of Trade Secrets*, 9 UCLA Pac. Basin L.J. 78 (1991). *See* FED. R. CIV. P. 92 (effective as of Jan. 1, 1998) (stating that the court will prohibit third parties from perusing or copying the record of proceedings upon a party's request and prima facie proof).

A. COPYRIGHT

1. *Copyright Protection for Software in Japan*

In 1982, for the first time, a Japanese court affirmed the availability of copyright protection for software,[111] and other courts followed this decision. In 1985, the Japanese Copyright Law (JCL) was amended to protect computer programs (the 1985 Amendment). During the amendment process, a legislative battle between the MITI and the Cultural Affairs Agency (CAA) of the Ministry of Education (MOE) developed—a *sui generis* approach versus a copyright approach.[112] The software industry supported the MITI's approach. The program right, as drafted by the MITI, was a hybrid of industrial property rights (such as patents) and copyright, but it is difficult to say whether the MITI's draft allowed for strong IP protection. At the least, it included the following pro-technology diffusion provisions: (a) shorter protection period (15 years from development); (2) a *saitei seido* (a compulsory non-exclusive license by arbitration in certain cases); and (3) the exclusion of moral rights. The MITI draft viewed software as an economic property with a short product cycle under rapid technological development, creating value only when used. Moreover, the draft mentioned the necessity of standardization of basic programs for interoperability.[113] The CAA, however, won the battle partly because of an international trend to protect software under copyright laws, partly because of pressure from the United States, and partly because of a string of Japanese court decisions affirming the copyrightability of computer software.

The 1985 Amendment added "program works" as a separate category to examples of copyrightable works listed in article 10 of the JCL.[114] Program is defined as "an expression of combined instructions given to a computer so as to make it function and obtain a certain result."[115] The definition of "program works" does not include design documents, flow charts, manuals, or other documentation that the JCL protects as lexical works (those involving novels, plays, essays, or lectures) or as pictorial works.[116]

---

111. Taito Corp. v. I.N.G. Enters. (Tokyo Dist. Ct. Dec. 6, 1982), *reported in* 482 HANREI TIMES, Feb. 1, 1983, at 70.

112. *See* JONATHAN BAND & MASANOBU KATOH, INTERFACE ON TRIAL 284-86 (1995); *see also* YOSHIKAZU TAKAISHI ET AL., SENTANSANGYU-NO CHITEKISHOYU-UKEN 33-42 (1990).

113. NOBUHIRO NAKAYAMA, SOFTWARE-NO HOUTEKIHOGO 214-29 (1988) [hereinafter NAKAYAMA, SOFTWARE].

114. Ms. Kumiko Bandou of the Ministry of Education, a senior official responsible for the 1985 Amendment, stated: "Several judicial precedents had justified programs as copyrightable works and as deserving protection under the Copyright Law. In order to make this clear, we added program works to the examples of copyrightable works. Kumiko Bandou, *An Amendment to the Copyright Law*, NBL 20 (1985) [hereinafter Bandou, *Amendment*].

115. Translation by Copyright Research and Information Center, Copyright Law of Japan (Nov. 1993 ed.) [hereinafter Copyright Center Translation].

116. The lexical work is similar to the term "literary work" in section 102(a) of the U.S. Copyright Act, but narrower. *See* BAND & KATOH, *supra* note 112, at 287.

The 1985 Amendment also defined the limits of copyright protection for software. Article 10(3) states:

> The protection granted by this Law [to program works] shall not extend to any programming language, rule or algorithm used for making such works. In this case, the following terms shall have the meaning hereby assigned to them respectively:
> (i) "programming language" means letters and other symbols as well as their systems for use as means of expressing a program;
> (ii) "rule" (*kiyaku*) means a special rule on how to use in a particular program a programming language mentioned in the preceding item;
> (iii) "algorithm" (*kaiho*) means methods of combining, in a program, instructions given to a computer.[117]

The term "rule" is commonly interpreted as concretely referring to interface specifications and protocols.[118]

Article 10(3) confirms a fundamental principle of an idea/expression dichotomy expressed in article 2: the JCL protects only expressions, not underlying ideas. Mr. Moriyuki Kado and Ms. Bandou, senior officials in charge of the 1985 Amendment, made this point clear. Mr. Kado stated:

> Copyrightable works mean, as defined in Article 2(1)-1, a production in which thoughts or sentiments "are expressed" in a creative way and which falls within the literary, scientific, artistic or musical domain. Therefore, *the languages used as a means of expressions as well as principles, ideas, and promises underlying expressions are not copyrightable works. Nor does the protection for copyrightable works extend to these.* This is right and proper (a logical consequence). However, regarding programs, the scope of copyright (especially rights of adaptation) occasionally becomes an issue in relation to an underlying idea, etc. Accordingly, article 10(3) was legislated in order to confirm the above point.[119]

This principle has been firmly established by judicial rule as well.[120] Thus, the JCL protects only literal expressions of software, while the Patents Law protects general inventive concepts.

2. *The Legality of Reverse Engineering*

Reverse engineering is a means of starting with a known product and working backward to define the process that aided in its development or manufacture.[121] An essential question is whether that intermediate copy constitutes copyright infringement. For reverse engineering, at least, a person needs to use a program on a computer. A temporary copy of the program is made in the computer's

117. Copyright Center Translation, *supra* note 115.

118. MORIYUKI KADO, CHOSAKUKENHO CHIKUJO KOUGI 98-99 (1994) [hereinafter KADO, KOUGI]; NAKAYAMA, SOFTWARE, *supra* note 113, at 42; BAND & KATOH, *supra* note 112, at 288.

119. *See* KADO, KOUGI, *supra* note 118, at 98-99 (emphasis added); *see also* Bandou, *Amendment*, *supra* note 114, at 20.

120. Dennis Karjala & Keiji Sugiyama, *Fundamental Concepts in Japanese and American Copyright Law*, 36 AM. J. COMP. L. 613, 649 (1988) [hereinafter Karjala & Sugiyama, *Fundamental Concepts*].

121. Kewanee Oil Co. v. Bicron, 416 U.S. 470, 476 (1974).

main memory. A memory dump may require an extra copy—a printout of the memory's contents including the program. Further, disassembly includes additional copying, translating the object code in memory back into an approximation of source code.[122]

3. *The Legality of Reverse Engineering under the Japanese Copyright Law*

As far as reverse engineering to achieve interoperability is concerned, a fair use defense may be extracted from the JCL: (1) the balancing of the authors' rights with the benefit of the public and technology development;[123] and (2) an idea/expression dichotomy.[124] Unlike traditional copyrightable works, software is a functional work and is an independent and separate category of copyrightable works under the JCL. More importantly, human beings cannot read and know its underlying ideas without obtaining an intermediate copy. Therefore, allowing reverse engineering necessary to achieve interoperability best serves the above two fundamental principles in the software context, as explained below.[125]

The JCL does not contain a general fair use provision, which the U.S. courts rely upon in analyzing a reverse engineering issue,[126] nor does the JCL directly address reverse engineering. However, article 1 provides, "The purpose of this [Copyright] Law is . . . to aim at protecting the rights of authors, etc., while considering *fair utilization* of these cultural products, and thereby to contribute to the development of culture."[127]

Furthermore, the Copyright Law protects only expressions. Underlying ideas should be available for the public use. If authors want to protect their underlying idea, their work must meet higher requirements for protection under the Patent Law (such as novelty and nonobviousness). Since the new 1993 Guidelines of the Agency of Patents clarified the availability of software patents, protection of ideas in software should depend upon patents.

Unlike traditional copyrightable works, not only is software inherently functional (as it directly causes machine processes to be performed), but people cannot study the underlying ideas of software—usually distributed only in electronic form—without first obtaining an intermediate copy. First, at a minimum, loading

122. *See* BAND & KATOH, *supra* note 112, at 17. (The reverse translation process of disassembly does not reconstruct certain explanatory "comments" and the names of "variables" of the original source code). *See Brief Amicus Curiae* of Eleven Copyright Law Professors in Sega Enterprises Ltd. v. Accolade, Inc., 33 No. 1 Jurimetrics 147, 152 [hereinafter *Brief Amicus Curiae*].

123. JCL art. 1.

124. JCL arts. 2 & 10(3).

125. *Cf. id.* at 154. (Also, it is much easier than in the U.S. Copyright Act where "literary works" also covers software. Under the U.S. Copyright Act, by emphasizing a functional or utilitarian aspect of software, "thinner" copyright protection for that aspect is rationalized).

126. Copyright Act, 17 U.S.C. § 107 (1994). *Cf.* JCL (containing several provisions that permit reproduction or exploitation of works in specific situations such as reproduction for private use).

127. JCL art. 1 (emphasis added). All translations of the JCL in this article are the author's, unless otherwise stated.

a program on to a computer and making a human readable copy in object code (1s and 0s) is necessary. Yet, a typical program in object code consists of thousands, sometimes millions, of 0s and 1s. It is virtually impossible to study a program by examining only its object code.[128] More realistically, a version closer to source code is necessary. Disassembly or decompilation is the only practical way one can read and analyze the ideas and functions contained in a computer program.

Thus, prohibiting all reverse engineering has the practical effect of completely denying access to the underlying ideas. This result contradicts the ideas/expression dichotomy. Total prohibition of software reverse engineering allows monopolization of ideas under lower requirements than the Patent Law seeks, and thus blocks the development of technology.

On the other hand, allowing all reverse engineering may erode incentive for developers, leading to less technological development. While encouraging interoperability may harm the industry giants' market share, it also may actually increase those programs' market share by enhancing the existing software. For example, one company's spell-checking program that is operable with a word-processing program of another company benefits the latter program's users and thus makes it more popular.

Developers of successful programs might license others to create interoperable software, but this method is problematic because: (1) those who do not accept unreasonable demands may not receive licenses; (2) time will be wasted on negotiations; and (3) software prices will be higher to account for such licensing fees. The crucial problem is the first one. For example, in the *Sega v. Accolade* case, Sega insisted on a policy that all programs developed by Accolade must be manufactured exclusively by Sega. Accolade gave up pursuing a license from Sega and instead turned to reverse engineering.[129]

Therefore, considering both interests, allowing reverse engineering—when necessary to achieve interoperability—serves the best balancing of authors' rights with the benefit of the public and technology development.[130] This is the purpose of the JCL.[131] By way of example, if ID code consisting of eight characters—necessary for interoperability—is copied into a new program, the legality of the new program will depend upon the copyrightability of the ID code. But the eight-character ID code is probably not copyrightable due to lack of creativity.

128. *Brief Amicus Curiae*, *supra* note 122, at 152-53.

129. Christian H. Nadan & James W. Morando, *Standardization and Interoperability Become Key Factor in Copyright Law*, 10 No. 4 COMPUTER LAW. 12 (1993).

130. Professor Nobuhiro Nakayama of Tokyo University, one of the leading intellectual property scholars in Japan, addresses the admissibility of reverse engineering. *See* NAKAYAMA, SOFTWARE, *supra* note 113, at 130-31. Mr. Masanobu Katoh, general manager of the Washington, D.C. office of Fujitsu, shares a similar view with the author. *See* BAND & KATOH, *supra* note 112, at 294-96.

131. *Cf.* JCL art. 30 (allowing reproduction for private use). Also with ownership of a reproduction of a program, the owner of a reproduction of a program work may make a reproduction or adaptation (including the reproduction of a derivative work created by the adaptation) to the extent necessary to use the program work on a computer. *See* JCL art. 47-2(1).

If the original program is copied more than is necessary for interoperability, it likely constitutes infringement. The copyrightability of the copied part into a newly developed program forms a key to the final judgment.

The 1985 Amendment's silence on reverse engineering did not intend to prohibit reverse engineering. On this point, an anonymous source who was heavily involved in the legislative process of the 1985 Amendment stated:

> Looking into the U.S. situation, the Copyright Act itself did not contain a specific provision addressing reverse engineering, while the Semiconductor Chip Program Act of 1984 did. Further, at that time [in 1984 and 1985], reverse engineering was so hot an issue that it was difficult to write [a specific article addressing reverse engineering]. On the other hand, *Article 10(3) could be interpreted to give room for allowing reverse engineering. Leaving this room, we postponed* deciding how to deal with reverse engineering. Actually, soon after the 1985 Amendment, a working group was formed to discuss the issue. A conclusion, however, was not reached.[132]

Indeed, in the early 1980s, no national copyright laws anywhere in the world dealt with software reverse engineering.[133] Competition in the software industry hinged on resolving this issue and protecting the non-literal elements of software. The debate in the United States focused on these issues.[134] During the course of Japanese legislation concerning software, the United States applied pressure, affecting the victory of the CAA's copyright approach over the MITI's *sui generis* approach.

Even inside the United States, opinions among the leading commentators were divided. The legality of loading a program into memory at the beginning of reverse engineering was not clear until 1988 when the Fifth Circuit allowed it under the fair use doctrine in section 107.[135] Disassembly, one method of reverse engineering, involves creating a derivative work. The solution of that issue had to wait until *Sega v. Accolade*, which upheld a fair use defense in reverse engineering "where disassembly is the only way to gain access to the ideas and functional elements embodied in a copyrighted computer program and where there is a legitimate reason for seeking such access."[136] Moreover, the above anonymous source stated that the legislators hoped that the 1985 JCL Amendments could be

132. Telephone Interview with Mr. Y (Nov. 24, 1995) (emphasis added) (Mr. Y is a career bureaucrat who shared his views with the author on the condition that he remain anonymous.). Because unlike in the United States, it is not congressmen and their staffs, but bureaucrats who work for the cabinets sending bills to the Parliament that mostly draft bills for legislation in Japan, their comments have more significance in Japan.

133. *See* BAND & KATOH, *supra* note 112, at xvii-xviii.

134. *See id.* at xviii.

135. Vault Corp. v. Quaid Software Ltd., 847 F.2d 255 (5th Cir. 1988).

136. Sega Enters. Ltd. v. Accolade, Inc. 977 F.2d 1510, 1527 (9th Cir. 1992). The case concerned a Sega lock-out program. Accolade copied Sega's software to discover the functional requirements for compatibility with Sega's console. The amount of copying was minimal (twenty bytes of initialization code and the letters of S-E-G-A), and there was no alternative. In considering the second statutory fair use factor, the nature of the copyrighted work, the court endorsed the *Computer Associates* decision. *See infra* note 164.

interpreted to allow for reverse engineering. Thus, the 1985 Amendment's silence on reverse engineering should not be construed to prohibit it.[137]

4. *Case Law: Did the Microsoft v. Shuuwa Decision of 1987 Prohibit Reverse Engineering?*

In *Microsoft Corp. v. Shuuwa System Trading KK*,[138] the defendant Shuuwa dumped Microsoft's object code from memory, disassembled the object code, and published the disassembled code in book form with Shuuwa's comments and labels.[139] Microsoft sought and obtained an injunction. This case is frequently cited in considering case law on reverse engineering, but a dispute exists as to whether it goes as far as prohibiting reverse engineering.[140] Although decided under the pre-amendment JCL, its precedential value remains because Japanese courts allowed protection of computer programs by copyright since 1982.[141]

The court held that Microsoft's program, as a part of the operating system, was protected by copyright. The court further held that converting object code into hexadecimal form constituted reproduction of the plaintiff's object code. It also stated that disassembling the object code and adding explanatory labels to the disassembled and interpreted version constituted reproduction because the work at issue and Shuuwa's work differed only by their explanatory labels. Finally, the court rejected the defendant's fair use defense, stating that "[P]ublishing the work at issue, which was not made public as a source program by its author . . . cannot be justified simply because the publication was done for the convenience of the users."[142]

---

137. 1219 HANREI JIHŌ 48; HANREI HYAKUSEN 60-61 (1994 ed.). Some commentators justify reverse engineering using article 47-2(1) of the JCL, which provides: [T]he owner of a reproduction of a program work may make a reproduction or adaptation (including the reproduction of a derivative work created by the adaptation) to the extent necessary to use the program work on a computer. *See* JCL arts. 47-52(1). This article, however, applies only to the "owner of a reproduction of a program work," not to a licensee. KADO, KOUGI, *supra* note 118, at 254. Because most of the industry practice uses licenses, article 47-2(1) actually covers few situations, and therefore does not solve the issues of reverse engineering. NAKAYAMA, SOFTWARE, *supra* note 113, at 78 n.2, "Because this Article is considered optional, contractual provisions govern when a contract exists."

138. (Tokyo Dist. Ct. Jan. 30, 1987).

139. The plaintiff, Microsoft, developed a BASIC interpreter as part of the operating system of NEC's PC-8001. Microsoft wrote the original program in ASSEMBLY language, converted it into object code, and stored it in the PC-8001's read-only memory (ROM). The source code of the program in ASSEMBLY language, which was the work at issue, was never published. The defendant, Shuuwa, converted the object code back into hexadecimal form and then disassembled it into ASSEMBLY language. Shuuwa converted the numerical addresses existing in the disassembled program into word labels for easier understanding. Then, Shuuwa published in book form the disassembled code together with labels and comments to assist readers' comprehension.

140. Professor Nakayama stated that the *Microsoft v. Shuuwa* decision ruled the entire series of acts of dumping, disassembly, and further publication illegal and that the scope of the *Shuuwa* decision did not cover reverse engineering. *See* NAKAYAMA, SOFTWARE, *supra* note 113, at 131-33.

141. *See supra* Part III.A.1.

142. HANREI HYAKUSEN, *supra* note 137, at 60.

The court in *Shuuwa* ruled that inevitable intermediate copying for reverse engineering is an unauthorized reproduction under the Copyright Law and, without justification, such copying is illegal. It clearly stated that both converting object code into hexadecimal form and disassembling the object code constituted a reproduction of the plaintiff's object code. The court analyzed each act of the defendant (including the defendant's dumping and disassembling) and found them to be unauthorized copying; this formed part of the court's logic leading to its conclusion, not mere dicta. In fact, the Administrative Department of the Japanese Supreme Court summarized the holding as follows: the act of disassembling object code in the ROM of a PC and putting labels and comments on it was judged to be copying of source code.[143] After stating that the defendant's dumping and disassembling constituted a reproduction, the court did not consider any justification for those acts. In sum, dumping and disassembling—inevitable in reverse engineering—were held to be unauthorized copying.[144]

Because of publication and massive reproduction in this case, few disagree with the conclusion of the decision; however, the court's ruling on intermediate copies is contradictory to the idea/expression dichotomy. Before the 1985 Amendment, the Japanese software industry was still in its infancy. Prepackaged software in the market was mostly game software.[145] The prevalent issue in Japanese courts at the time concerned the copyrightability of software. Considering the stage of development of both the software industry and the case law regarding software in the early 1980s, it is not surprising that the court could not offer a detailed discussion of software reverse engineering. The current JCL, however, contains provisions for software. For example, article 10(3) provides an idea/expression dichotomy in the software context. In 1995, after external pressure halted the CAA's legislative effort to clarify the issue, observers awaited relevant court decisions.

5. *Halted Japanese Reverse Engineering Initiative*

In the mid-1990s, CAA sought the adoption of a provision similar to the EC Directive that allows reverse engineering at least to the extent required to achieve interoperability,[146] but this effort was halted by outside pressure from the United States.[147] The CAA's effort started in July 1993 by issuing a press release stating that its Consultative Committee would study: (1) whether the JCL should be revised to provide for reverse engineering; (2) whether limitations of rights should

143. GYOOSEISAIBANREI GAIKAN 778 (1993).

144. Mr. Keiji Sugiyama and Mr. Yoshikazu Takaishi expressed a similar view. HANREI HYAKUSEN, *supra* note 137, at 61. TAKAISHI, *supra* at 112.

145. *See* Correspondence with Professor Katsuya Hirose, Hosei University (1995) (on file with author).

146. The EC Directive of May 14, 1991 allows reverse engineering when necessary to achieve interoperability between programs.

147. *See* BAND & KATOH, *supra* note 112, at 314-15.

be expressly stipulated; and (3) what sort of limitations and conditions would be appropriate.[148] The press release justified the study by pointing to the EC Directive and the U.S. court decisions that ruled copying associated with reverse engineering is a fair use.

In September 1993, the Japanese Federation of Economic Organizations (Keidanren), which is similar to the U.S. Chamber of Commerce, submitted its comments to the Consultative Committee (Keidanren Submission).[149] This Keidanren Submission reflected the opinion of computer hardware manufacturers, not software companies. Keidanren is a business association of traditional and large companies. Neither of the two software trade associations (JISA and JPSA) commented on this matter. Although the Intellectual Property Rights Committee of the JISA discussed the issue, its term expired before reaching a conclusion. The Committee was not requested to write a formal report because the matter had become a political and trade issue, and the content of the Committee's discussion was not made public (as usual).[150] The JPSA did not comment on the CAA's attempt because it thought it premature to issue its decision on the issue of reverse engineering.[151] The CAA's press release and the Keidanren Submission elevated an obscure intellectual property issue to a trade-related showdown.

Even inside the United States, opinions on reverse engineering had been divided between proprietary companies and open system companies. For example, the American Committee for Interoperable Systems to Commerce sent a letter to Secretary Ron Brown dated November 17, 1993 stating:

> Most methods of software reverse engineering, other than disassembly, are noncontroversial within the computer industry. . . . Although some may try to draw a dividing line between the U.S. and Japan on this issue, the actual dividing line is between primary proprietary companies . . . and open system companies.[152]

Eventually, however, the U.S. Government took a stance favorable to the proprietary vendors, and put political pressure on the Japanese Government to quash the CAA's effort.[153]

In May 1994, the Consultative Committee issued its final report that stated that no conclusion on specific revisions of the Copyright Law at that stage could

148. *Id.* at 297.

149. It supported disassembly regardless of its purpose. In addition to the U.S. court decisions and the EC Directive, Keidanren justified its position by arguing that: (1) analysis of computer programs is crucial for the advancement of science and beneficial to society and as such prevents "redundant investments in similar technologies;" (2) reverse engineering is generally accepted in all industries; and (3) prohibiting reverse engineering would result in copyright protection for ideas properly protectable only by patent and trade secret law. *See* BAND & KATOH, *supra* note 112, at 298.

150. *See* Correspondence with a member of the Intellectual Property Rights Committee of JISA (June 16 & 17, 1994) (on file with author).

151. Interviews with members of the Illegal Copy/Legal Protection Joint Committee of JPSA (Mar. 1995).

152. *See* BAND & KATOH, *supra* note 112, at 299-307

153. *See id.* at 299-312.

be reached and, as a counter-argument to the U.S. Government's position, asserted that the level of software protection in Japan was substantially the same as that in Europe and the United States.[154] Thus, the CAA's effort to clarify the reverse engineering issue was halted and frozen.

6. *Structure, Sequence, and Organization (SSO) and User Interfaces*

a. Processing Flow

In *System Science Corp. v. Toyo Sokki KK* (1989), a case involving the algorithm limitation on copyright protection under the JCL,[155] the Tokyo District Court and the influential Tokyo High Court held that processing flow in a computer program is not protected by the program's copyright.[156]

The plaintiff, System Science, developed four programs for biochemical measurements. The defendant admitted copying three programs, except one known as CA-7 II, and installing them into ROM format for marketing. The defendant marketed four competing products, each containing a program called CA-9. The major issue was whether CA-9 was an adaptation of CA-7 II. The High Court stated:

> To decide that a program at issue infringes the copyright in a program work, there must be creativity in the combination of instructions of the program work. Further, the combination of the program developed later must be similar to the creative part of the program work . . . [S]ymbols that express a program are very limited, and their system (grammar) is also rigorous. Therefore, in aiming at functioning a computer to obtain a result more effectively, not a few portions of the combinations of instructions cannot but be similar. Accordingly, regarding program works, we must determine copyright infringement with caution.[157]

Creativity does not mean the higher originality standard that German courts required before the adoption of the EC Software Directive.[158] It means that original, but highly mundane expressions may be unprotected by the JCL.[159] Professor Karjala explains, "The analysis might sound more coherent to a Western ear if undertaken in terms of the idea/expression distinction, coupled with such doctrines like the merger of idea and expression and *scenes a fair*."[160]

The High Court analyzed the issue as follows:

> Hardware performs the functions of measurement mode change, keyboard input, allotment of measurement area and write-in on common memories; what CA-7 II or CA-9

154. *See id.* at 313-14.

155. *See* Dennis Karjala, *Copyright Protection of Computer Software in the U.S. and Japan: Part II*, 7 EUROPEAN INTELLECTUAL PROPERTY REVIEW 231, 232 (1991) [hereinafter Karjala, *Japan*].

156. Tokyo High Court decision of June 20, 1989. 1322 HANREIJIHOO 138; For the details of the Tokyo District and High Court decisions, see Karjala, *Japan*, *supra* note 155, at 235.

157. 1322 HANREIJIHOO 140.

158. BAND & KATOH, *supra* note 112, at 291. "[I]t . . . simply means that copyright would extend only to the unconstrained elements of a work." *Id.*

159. Karjala, *Japan*, *supra* note 155, at 231-32.

160. *Id.*

> performs is only a printing part; the combination of instructions for the processing routines after data entry from the processor cannot but be the same due to constraint by the hardware; CA-7 II and CA-9 adopt a common (standard) combination of instructions for the processing routines while waiting for the printer; common sense tells the 4100H partition to set the subroutine stack because of the RAM area of [defendant's] hardware, and; *'processing flow' in a computer program is algorithm, 'kaihoo' under Article 10(3)-3 of the JCL, which is not protected by copyright.*[161] Therefore, creativity cannot be found in the part of CA-7 II at issue. Further, while CA-7 II has 12,000 bytes, CA-9 has 763 bytes. Moreover, what System Science asserts as similar between the two programs are minimal bytes.[162]

Accordingly, the court concluded that the defendant's CA-9 was not an adaptation of CA-7 II.

Regarding a program that was not literally copied, the Tokyo High Court opinion showed sharp contrast to the Third Circuit's decision in *Whelan Inc. v. Jaslow Dental Laboratory, Inc.*,[163] which protected SSO. The Japanese courts took a less protective approach, similar to the 1992 decision by the Second Circuit in *Computer Associates Int'l v. Altai, Inc.*,[164] which criticized *Whelan* and was followed by many other courts.[165] The *Computer Associates* decision excluded broader structural features of software from copyright protection than the *Whelen* decision. The issue was whether, and to what extent, the non-literal elements of software—such as general flow charts and parameter lists—were protected by the U.S. Copyright Act. The Second Circuit held that a program must be carefully separated into copyrightable and uncopyrightable components and that scrutinized copyrightable parts should be compared with the structure of an allegedly infringing program. Further, the Second Circuit narrowed the scope of copyrightability of software features. Acknowledging the essentially utilitarian nature of software, it ruled that copyright did not protect elements dictated by efficiency or external factors, including hardware specifications and compatibility requirements of other programs.

---

161. "The [Tokyo High] court does not clarify just what it means by 'processing flow,' but the term sounds very much like what the trial court in *Whelan* defined as expression." *Id.*

162. 1322 HANREIJIHOO 140 (emphasis added).

163. 797 F.2d 1222 (3d Cir. 1986), *cert. denied*, 479 U.S. 1031 (1987).

164. Computer Assocs. Int'l, Inc. v. Altai, Inc., 982 F.2d 693 (2d Cir. 1992). As of 1990, Professor Karjala wrote:

> [The *System Science*] decision should be contrasted with Whelan and Apple Computer v. Franklin Computer, 714 F.2d 1240 (3d Cir. 1983)] decisions in the United States . . . The *System Science* approach is more closely akin to the analysis of NEC Corp. v. Intel Corp., 10 USPQ 2d 1177 (N.D. Cal. 1989), which denied infringement based on similarities arising out of compatibility or efficiency considerations.

Karjala, *Japan, supra* note 155, at 238.

165. *E.g.* Atari Games Corp. v. Nintendo of Am. Inc., 975 F.2d 832 (Fed. Cir. 1992); Sega Enters. Ltd. v. Accolade, Inc., 977 F.2d 1510 (9th Cir. 1992). National Educ. Support Sys., Inc. vs. Autoskill, Inc., 994 F.2d 1476 (10th Cir. 1993); Gates Rubber Co. v. Bando Chem. Indus., Ltd., 9 F.3d 823 (10th Cir. 1993); Engineering Dynamics, Inc. v. Structural Software, Inc., 26 F.3d 1335 (5th Cir. 1994) (user interface case).

b. User Interface

User interface is the design of the video screen and the manner in which information is presented to users and by which users interact with computers. Most significantly, it encompasses displays and commands appearing on the screen.

With regard to user interfaces, Japanese courts protect not only a game program as a program work, but also its screen displays as a cinematographic work.[166] It remains unclear whether courts will extend copyright protection for screen displays beyond the realm of game programs to more functional software. Concerning user interface, little case law exists in Japan. In the 1990s, there have not been any significant copyright lawsuits in Japan involving software, other than recent district court decisions centered on the moral right issue in game software.[167] Because the United States does not have a general counterpart to moral rights, those decisions are not explored here.

In the United States on the other hand, the Ninth Circuit held in *Apple Computer, Inc. v. Microsoft Corp.* that the graphical user interface (GUI) of Microsoft's windows environment did not infringe Apple's copyrights for the user interface of its Macintosh OS.[168] The court found that most of the allegedly infringing features of Windows 2.03 and 3.0 were either authorized copying under a license agreement or unprotected elements.[169] Then, in *Lotus v. Borland*, the First Circuit ruled that Lotus 1-2-3's menu command hierarchy was not copyrightable subject matter because it was a method of operation unprotected under the U.S. Copyright Act.[170] Because the U.S. Supreme Court affirmed the First Circuit decision in a 4-4 per curiam decision,[171] the First Circuit decision stands, but the status of the issue remains unclear.

From the foregoing, it follows that the JCL and case law have provided relatively sufficient protection for software since the 1980s. However, the U.S. courts stretched the scope of protection by copyright law too broadly, such as in *Whelan v. Jaslow*, and then later narrowed the scope. In Japan, the courts did not stretch copyright protection to SSO; admissibility of reverse engineering remains uncertain although legislative bases exist in the author's opinion. Case law does not exist regarding copyrightability of user interfaces, except screen displays of game

166. Kumiko Bandou, *Video Game-no Eizoo-to Program*, HANREI HYAKUSEN 68-69 (1994 ed.). *See, e.g., Digdug* case (Tokyo Dist. Ct. Mar. 8, 1985).

167. *Sangokushi* case (Tokyo Dist. Ct. July 14, 1995) (on appeal with decision expected March 1999), 1538 HANREI JIHŌ 203; *see also NEOGEO Game Machine* case (Osaka Dist. Ct. July 17, 1997); *Tokimeki Memorial* case (Osaka Dist. Ct. Nov. 27, 1997) (on appeal with decision expected February 1999).

168. *See* Apple Computer, Inc. v. Microsoft Corp., 35 F.3d 1435 (9th Cir. 1994), *cert. denied*, 115 S. Ct. 1176 (1995).

169. *See id.* at 1435, 1442.

170. *See* Lotus Dev. Corp. v. Borland Int'l, Inc., 49 F.3d 807, 816 (1st Cir. 1994).

171. *See* Lotus Dev. Corp. v. Borland Int'l, Inc., 516 U.S. 233 (1996).

programs. On the other hand, U.S. circuit court decisions have narrowed the scope of software protection by copyright. That is, protection stretched to SSO was narrowed; a reverse engineering was upheld as a fair use, and user interfaces such as a GUI and a menu command hierarchy were judged not copyrightable.[172]

Thus, Japanese copyright law is not significantly less protective of software when compared with that of the United States. Rather, software protection in the United States is weaker as far as reverse engineering is concerned. If software developers seek stronger protection for what is not protected by copyright, they should pursue protection under patents by meeting higher standards.[173] Further, in 1997, to meet the current network era, the JCL was amended both to expand an already-existing author's right concerning interactive transmission[174] and to make the copyright of a computer program include wire transmission to the public on the same premises (such as via a local area network or LAN).[175]

B. Patents on Software-related Inventions in Japan

Interests in patents on software-related inventions (software patents) have been increasing in both Japan and the United States. The Japanese Patent Law states that patents are exclusive rights to commercially license the patented inventions.[176] While a copyright is established when a copyrightable work is created, patents are granted to applicants after the controlling authority follows certain procedures. Recently, new guidelines for software-related inventions were published in both countries. After briefly explaining the development

172. As a whole, an evident trend in the United States is for courts to recognize certain "externalities" such as compatibility, *see, e.g., Computer Assocs.*, 982 F.2d at 710, Lotus Dev. Corp. v. Borland Int'l Inc., 49 F.3d 807, 817 (1st Cir. 1995); functionality, *see, e.g., Computer Assocs.*, 982 F.2d at 709-10; and industry standarization, *see, e.g.*, 982 F.2d at 710, Apple Computer, Inc. v. Microsoft Corp., 35 F.3d 1435, 1438 (9th Cir. 1994), Brown Bag Software v. Symantec Corp., 960 F.2d 1465, 1472 (9th Cir.), *cert. denied*, 113 S. Ct. 198 (1992), as limiting the scope of protectable elements of computer programs. Further, some recent decisions applied the fair use doctrine more broadly to cases involving the copying of functional elements of a program to achieve compatibility. *See* David Hayes, *A Comprehensive Current Analysis of Software "Look and Feel" Protection, in* THE 9TH ANN. ADVANCED COMPUTER L. INST. vol. II at 515, 673, & 676 (Georgetown University Law Center, March 14-15, 1996).

173. Also, possible protection for user interface should be pursued under design patents. For example, Microsoft obtained a design patent for a specific combination and order of about 160 icons. (U.S. Patent No.: Des.341,848; Date of Patent: Nov. 30, 1993). Others can use those icons, but cannot obtain a design patent on the same combination and order of the icons, according to Stephen L. Peterson, Esq. of Finnegan, Henderson, Farabow, Garrett & Dunner in Washington, D.C.

174. Japanese Patent Law art. 23 [hereinafter JPL] ((1) The author has the exclusive right to make the public transmission of his work (including making his work transmittable in the case of interactive transmission). (2) The author has the exclusive right to publicly communicate, by means of a receiving apparatus, his work whose public transmission has been made.)

175. *See* JPL art. 2 (vii*bis*).

176. *See* JPL art. 68. On the other hand, in the United States, patents are defined as the right to exclude others from making, using, offering for sale, or selling the invention throughout or importing invention. 35 U.S.C. § 154 (1994).

of software patents in Japan, the scope of such patents in Japan and the United States will be compared below.

1. *Patents on Software-related Inventions in Japan*

In Japan, a subject matter must be an "invention,"[177] and have utility (industrially applicable), novelty, and nonobviousness[178] to be patented. Invention is defined as "the highly advanced creation of technical ideas by which a law of nature is utilized."[179] Japan uses a first-to-file system, as do most industrialized countries, excluding the United States. Novelty is lost if the invention is publicly known or worked in Japan, or described in a publication distributed in Japan or elsewhere prior to the filing date of the patent application.[180]

In 1995, in response to the General Agreement on Tariffs and Trade (GATT), Japan extended its patent term to twenty years from the filing date. Also, a post-grant opposition system was substituted for a pre-grant opposition system.[181] Then, on February 24, 1998, the Japanese Supreme Court upheld the Doctrine of Equivalents in the so-called *Ball Spline Bearing* case.[182] Further, in 1998, the Japanese Patent Law was amended to strengthen the system for awarding damages.[183] Before the amendment, damages were often calculated based on a normal license fee due to difficulty in proving an infringer's net profit, which the court considers in estimating damage awards.[184] In contrast, the new law enables a patent holder to prove its damages based on proof of the sales volume of the infringer and provides for consideration of specific circumstances such as the parties' competition, regardless of a normal license fee between bona fide contractors.[185]

177. JPL art. 2.

178. JPL art. 29. The requirement level of utility is higher than that in the United States. Telephone Interview with Ken-ichi Hattori, Esq., former JPO Examiner (Mar. 19, 1996).

179. JPL art. 2-(1).

180. JPL art. 29-(1).

181. The effective date of the extended patent term was July 1, 1995 and the post-grant opposition system was January 1, 1996.

182. *Heisei 6 (o) No. 1083/1994*. The factors so that the Doctrine of Equivalents is approved (that is, infringed) include: (1) a different part between the accused art and the patented invention is not an essential part of the patented invention; (2) substitution of the different part of the accused art for that of the patented invention, which still performs the same function and has the same effect as the patented invention; (3) such substitution could be easily conceived by someone of ordinary skill when the accused art was manufactured; (4) the accused art was not known nor easily conceived by someone of ordinary skill at the time of the patent application; and (5) there was no special situation (for example, the accused art was intentionally eliminated from the patent claim during the patent prosecution).

183. Law No. 51 (1998) (Japan) (entered into force Jan. 1, 1999).

184. Yasukazu Irino, *Tokyohouno Ichibuwo Kaeseisuru Houritsuno Gaiyo*, NBL 643 (1998).

185. Japanese Patent Law art. 102 (1999). However, punitive damages are not awarded under the Japanese law and the Japanese Supreme Court refused to enforce punitive damages awarded by a foreign court in *Northcon v. Mansei Kogyo* (1997).

In the late 1980s, the software industry was surprised by the "YES" patent infringement case, even though they had been interested in patents on software-related inventions.[186] YES Corporation, which managed the patent on equipment for budget and storage management, sued twelve companies for patent infringement.[187] Several defendants were not manufacturers, but software companies.[188] The patented invention related to "an apparatus for financial management, inventory control or the like comprising a display unit, an input unit, a memory having various files, an output unit and processing unit having five processing means."[189] The case settled, leaving the issue ambiguous.[190]

Yoshikazu Tani, a Tokyo patent attorney states, "Software patents had already been granted on systems such as the YES patent by the Japanese Patent Office (JPO). The YES patent was not a surprise."[191] While this may have been the case among patent practitioners, many in the software industry were surprised. This led to the JISA publication, *Software Related Prior Art Glossary*, to help the JPO build a prior art database on software-related technology when the JPO announced the new guidelines in 1993, as discussed below.[192]

a. Japanese Patent Office Guidelines in 1993 and 1997

The practice of the JPO has more significance than that of the U.S. Patent and Trademark Office (PTO) because little case law exists in Japan.[193] The demand for protection of computer software by patents has increased in the 1990s,[194] and forced the JPO to introduce in June 1993 its new *Examination Guidelines for Computer Software Related Inventions* (1993 Guidelines).[195] The 1993 Guidelines unified the three coexisting standards/guidelines—(1) Examination Standard for Computer Program Related Inventions (Part I) (December 1975); (2) Examination Guidelines for Inventions Relating to Microcomputer-Applied Technology (December 1982); and (3) Examination Method of Computer Software Related Inven-

186. Telephone Interview with Akira Uchinuno, *supra* note 15.

187. Akira Uchinuno, *Legal Protection of Software/Algorithm, in* JAPAN-U.S. SYMPOSIUM ON THE FUTURE OF THE PROTECTION OF SOFTWARE AND ALGORITHMS—A DIALOGUE BETWEEN ENGINEERING AND LEGAL COMMUNITY 246 n.16 (Mar. 1995).

188. *Id.*

189. YOSHIKAZU TANI, THE PROTECTION OF COMPUTER SOFTWARE IN JAPAN 14 (AIPLA 19th Mid-Winter Inst. Working Paper, Jan. 24-27, 1996) [hereinafter TANI]; Uchinuno, *supra* note 187, at 246. The patented equipment was to input several basic data on budget and storage through a common format on the screen, to fill in and update master files and data files, and to create books and graphs for management by reading necessary data from those files.

190. *Id.*

191. Telephone Interview with Yoshikazu Tani (Feb. 26, 1996).

192. Telephone Interview with Akira Ogata of JISA (Feb. 29, 1996). The Japanese title is *Software Shuuchi Kanyoo Gijyutsushu.* The other reason was the request of the JPO.

193. HIDETAKA AIZAWA, SOFTWARE KANRENHATSUMEI-NO TOKKYOHO-NIYORU HOGO-NO GENJOO 73 (SOFTIC 1995).

194. TANI, *supra* note 189, at 2.

195. JAPANESE PATENT OFFICE, EXAMINATION GUIDELINES FOR INVENTIONS IN SPECIFIC FIELDS 1 (1993) [hereinafter 1993 GUIDELINES].

tion (Draft) (January 1989)—to employ a new concept from the standpoint of information processing and hardware resources.[196] The first standard was a general one dealing mostly with method claims.[197] The second guideline dealt with apparatus claims relating to a chip microcomputer.[198] Under this guideline, for example, *kana-kanji* conversion method[199] was granted a patent.[200] The third method, which supplemented the first standard and the second guideline,[201] responded to the increase in granting of patents on software-related inventions that would have been rejected under strict application of the first standard and the second guideline.[202] Under the method, many software-related inventions obtained patents.[203] All of the three standards/guidelines adopted the concept of "cause and effect relationship."[204]

The 1993 Guidelines instruct: (1) a claim should be considered as a whole in judging whether it is statutory subject matter; and (2) if the invention as a whole either (i) utilizes a law of nature in the information processing by the software, or (ii) utilizes hardware resources, then the invention is considered to utilize natural laws.[205] The 1993 Guidelines also instruct that description as a product category or a process category is not relevant to determining the utilization of a law of nature.[206] The 1993 Guidelines give some examples of statutory inventions, which were made by modifying patents granted in the past,[207] and state that a programming language and a program per se are not patentable.[208]

In February 1997, the JPO issued a new guideline accepting computer-readable medium with computer programs as a valid patent claim (1997 Guidelines).[209] Thus, the holder of such patent may now assert direct infringement against the sales of floppy disks and CD-ROMs with such programs, whereas only an assertion of indirect infringement was possible under the 1993 Guidelines.[210] Thus, protection of software-related inventions has been strengthened.

196. *Id.*

197. *Id. See also* Telephone Interview with Akira Uchinuno, *supra* note 15.

198. *See* 1993 GUIDELINES, *supra* note 195; *see also* Telephone Interview with Akira Uchinuno, *supra* note 15.

199. Conversion method from an input in Japanese alphabet (*Kana*) to Chinese letters (*Kanji*). Japanese sentences consist of Chinese letters and Japanese alphabets.

200. S58-21287, Kenji Ushihisa, *Patentable Subject Matter in Japan*, SOFTWARE KANRENHATSUMEI-NO TOKKYOHO-NIYORU HOGO-NO GENJOO 8 (SOFTIC 1995).

201. TANI, *supra* note 189, at 2.

202. Ushihisa, *supra* note 200, at 7.

203. *Id.* at 9.

204. TANI, *supra* note 189, at 2.

205. *Id.* at 6.

206. Ushihisa, *supra* note 200, at 9.

207. Telephone Interview with Yoshikazu Tani, Esq. (Mar. 20, 1996).

208. 1993 GUIDELINES, *supra* note 195, at 6-7.

209. *Computer Software Related Inventions*, IMPLEMENTING GUIDELINES FOR INVENTIONS IN SPECIFIC FIELDS. This guideline applies to patent applications on and after April 1, 1997. *See* <http://www.jpo-miti.go.jp/guide/soft-e.txt.>

210. *Software—kanren Tokkyoni Kansuru Chousakenkyuu Houkokusho* 109 (SOFTIC, Mar. 1998).

After the publication of the 1993 Guidelines, applications for software-related inventions increased dramatically.[211] In Japan, substantive examination commences only upon request for examination,[212] which must be filed within seven years from filing of the application.[213] Examinations are requested on about sixty percent of the applications.[214] On average, such an examination takes approximately two years to reach the final action of examiners.[215] At that pace, several years must elapse in order to observe how the 1993 and 1997 Guidelines will affect the examination process.

b. Comparison between Japan and the United States

In comparing trends in patent protection covering software-related inventions in Japan and the United States, an interview with an experienced international practitioner is an invaluable source because little has accurately been written. Yoshikazu Tani, Esquire notes:

> The scope of patent protection for software-related inventions was not necessarily narrower in Japan than in the U.S. A long time ago, protection was wider in Japan. While *Diamond v. Diehr* [1981] was disputed up to the U.S. Supreme Court, Japanese Patent Office granted patents without any trouble on similar inventions including mathematical algorithms. Another example is a data structure in files of memory. It has been patentable subject matter for a long time in Japan; in the U.S., it was not previously accepted. However, the U.S. Patent Office (PTO) has recently become more supportive of patents on software-related inventions. Very recently [*In re Lowry* (1994) and the 1996 PTO Guidelines], the U.S. PTO has started to accept a data structure in files of memory as patentable subject matter if the data structure is stored on medium. Concerning computer-readable medium with computer programs, the U.S. PTO and JPO rejected them before. Yet, the U.S. PTO has changed its position to accept computer-readable medium with computer programs under the 1996 PTO Guidelines [incorporating *In re Beauregard* (1995)].[216]

Soon after Mr. Tani made these comments, the JPO began to accept computer-readable medium with computer programs as statutory inventions pursuant to the 1997 Guidelines.

Thus, determining which country offered a wider scope of patent protection for software-related inventions during the 1980s is not a simple task. Resembling a Venn diagram, each country's scope of the patent protection overlapped that of the other. By the mid-1990s, the United States clearly offered wider patent

211. More than 10,000 applications are said to be filed per year. Uchinuno, *supra* note 187, at 246 n.16 (position paper for the Japan-U.S. Symposium on the Future of the Protection of Software and Algorithms).

212. JPL art. 48-2.

213. JPL art. 48-3.

214. Telephone Interview with Mr. Hirotsuna Yamashita, Institute of Intellectual Property (Wash. D.C. office) (July 1995).

215. Telephone Interview with Mr. Hirotsuna Yamashita, Institute of Intellectual Property (Wash. D.C. office) (Feb. 1996).

216. Telephone Interview with Yoshikazu Tani, *supra* note 207.

protection for software-related inventions than Japan, but Japan is now following in a similar direction.

The foregoing Part III shows that the Japanese laws have provided relatively sufficient protection[217] for software since the 1980s. Japanese protection has not been significantly different from that in the United States, although the scope of U.S. copyright and patent protection occasionally changed to a large extent due to active litigation. Also, both the CAA's triumph over the MITI's *sui generis* approach, and the software industry's silence during the CAA's halted effort in 1994 concerning reverse engineering show, in large part, that the Japanese software industry has not affected the legal scheme of copyright protection for software.

IV. Conclusion

Though most of the comparatively larger software companies in the dominant custom segment are *keiretsu* software companies, they have not depended upon implicit contracts with customers within the same *keiretsu* group since the 1980s. Even when explicit contracts are executed in Japan, legislation has great significance due to the relative simplicity of contracts and the lack of case law. This is further evidenced by moves of the trade association of custom software companies in the mid-1980s toward legislation protecting software. Thus, the dominance of the custom segment did not necessarily result in weak legislative IP rights protection.

Significantly, *keiretsu* software companies, most of which are players in the custom software segment, do not suffer from a financing problem, thanks to their creditworthiness, which is based on the social recognition that a key company/main bank of the same *keiretsu* group will not let them go bankrupt. Although strong financial backing is crucial to innovation, especially in the investment-first and R&D-driven prepackaged software business, Japanese software companies have difficulty in obtaining financing due to poorly developed venture capital, a lack of a real NASDAQ-type market, and bank lending that focuses on real property as collateral. This difference in financial strength has created the discrepancy between the dominant custom segment and the underdeveloped prepackaged software segment.

Moreover, the *keiretsu* structure does not always bring sufficient business to its member software companies. Further, with the ongoing recession, Japanese companies pay more attention to current profits and cost performance than to trade relationships with their affiliated *keiretsu* companies. Consequently, *keiretsu* software companies face severe competition both inside and outside the

217. *See also* Brian G. Strawn, *Guide to Japanese Intellectual Property Law*, 26 AIPLA Q. J. 58 (1998) ("Japanese law provides extensive protection of intellectual property rights, including patents and utility models, trademarks and service marks, copyrights, trade names, and industrial designs.").

keiretsu. Still, they remain within the *keiretsu* due to their creditworthiness as *keiretsu* companies, but continue to diversify their trade partners, more to the outside of the *keiretsu*.

Japanese laws have provided sufficient protection for software since the 1980s, and have not been significantly different from the scope of protection in the United States. This has been the case with copyright, which played a major role in protection of software until recently. Both countries' scope of patent protection for software-related inventions overlapped with the other's in the 1980s. Though the United States had trended toward wider patent protection for software-related inventions than Japan by the mid-1990s, Japan has been following in a similar direction. However, Japanese IP protection for software did not help Japanese software companies to attract investment capital.

From the foregoing, the author mostly disagrees with Merges' hypotheses regarding the Japanese software industry. Concerning a causal link among legislative IP rights protection in Japan, the underdeveloped prepackaged software segment, and the *keiretsu* structure, this author formulates that legislative IP protection did not help Japanese software companies to obtain strong financial backing, which is especially crucial to enter and remain innovative in the investment-led and competitive prepackaged software business. *Keiretsu* software companies, which dominate the larger custom software segment, obtain financing easily due to their creditworthy *keiretsu* status. This difference has led to the superiority of the custom segment over the prepackaged segment in Japan.

Appendix

Keiretsu Influence on Transactions Costs and Financing Costs

1. *A's transaction costs to deal with B*

Trading partners A–B		(1) Information gathering costs	(2) Bargaining costs	(3) Monitoring costs
K1–K2 within the same *keiretsu*	[past]	low	low	low
	[present]	low*	It depends upon bargaining power, etc.	Not low, especially re a large corp. K2 *Still low re a software corp. K2*
KX–KY different *keiretsu* or *non*-K–K	[past]	—	high	Low. KX or non-K can depend upon the monitoring by KY's or K's main bank.
	[present]	—	It depends.	Not low re a large corp. *Still low re a software corp. KY*; K
any corp.–non-K	[past & present]	—	It depends.	—

K: *keiretsu* company
non-K: not a *keiretsu* company
—: no effect
* If bank lending is low, information gathering through banks (not through the other *keiretsu* companies) decreases; information gathering costs are not necessarily low. However, if B is a software company, low costs are still the case. "Bargaining costs" here means time and labor spent for contractual negotiations.

Though monitoring costs remain low as far as software companies are concerned, the above change in bargaining costs has lessened the difference between *keiretsu* and non-*keiretsu* companies. Further, as a flip side to the very recent decrease of cross-shareholdings and more focus on current cost performance as stated above, costs to monitor trade partners are supposed to have increased within the *keiretsu*.

2. Keiretsu *Companies Financing Costs*

Low—easy to obtain financing because of creditworthiness supported by the implicit social recognition that key companies of the *keiretsu* will not let a *keiretsu* company go bankrupt.