COMPUTER SOFTWARE AND STRICT PRODUCTS LIABILITY

Computerization is clearly the wave of the future in both industry and the home. Unfortunately, along with the benefits of automation, society faces a new source of potential injury. One means of resolving the problem of injuries due to defective computer programs would be to allow the victims to bring an action under strict products liability. However, due to the characteristics of a computer program, it may be difficult for a court to classify the program as a "product." This Comment examines this issue and discusses some possible solutions.

INTRODUCTION

"First Law— A robot may not injure a human being, or, through inaction, allow a human being to come to harm,
Second Law—A robot must obey the orders given it by human beings except where such orders would conflict with the First Law . . . ."

Three Laws of Robotics, by Isaac Asimov[1]

The world has now witnessed its first human death at the hands of a robot.[2] With the rapid increase of computer use in both government and industry, such an event was probably inevitable, and it is likely that similar accidents will occur in the future. The potential for human injury due to a malfunctioning computer program is tremendous. No longer can computer-caused injuries be relegated to the realm of futuristic science-fiction novels.

---

1. ISAAC ASIMOV, I, ROBOT 6 (1950).

2. The *New York Times* reported that a 37-year-old factory worker was crushed to death by an industrial robot at the Kawasaki factory near Kobe, Japan. According to government investigations, the worker entered a restricted zone where machines were in operation. He evidently saw something wrong with one of the machines, began fixing it, and became "so engrossed with fixing the machine that he did not notice the approach, from behind, of a transport robot that delivered parts to the machine." N.Y. Times, Dec. 13, 1981, § 3, at 27, col. 1.

Although the accident was apparently brought about by the worker's own negligence, there was some thought that it might have been caused by a defect in the computer program.

The robot-caused death mentioned above is not the only case where an error in a computer program has been investigated as the basis of an accident. In the aeronautics industry, where computer use is widespread, program error has been the suspected cause of several airline crashes or difficulties.[3] In the medical field, extensive computer use could lead to an increased number of injuries,[4] as doctors are relying more and more on computers to aid in their analysis of patients.

For a demonstration of the potential problems involved, consider a hypothetical example. Some factories already use computer-controlled robots to perform lifting and carrying jobs. If the computer program that controlled one of the robots was defective—either because the original program was created incorrectly or because an error occurred in the program instructions when it was copied—the robot could be dangerous. If even one of the tiny bits of information that form the program's instructions was incorrect, the "forklift" robot could drop its load before reaching its destination, perhaps injuring someone in the process. If the same robot was equipped with a computerized sensor device to stop it when people crossed its path, injury might occur if the system failed to work properly. There could be nothing wrong with the sensor device, or the actual mechanical part of the machine. However, if because of an incorrect program the machine failed to stop even after it had received the information from the sensor, an unsuspecting worker or bystander could be hurt.

Today's computer-related damages in commercial and economic areas are generally handled under negligence law.[5] If negligence

---

3. *Crash Spurs Fixes to F-18*, AVIATION W., Dec. 15, 1980, at 24. See Gemignani, *Product Liability and Software*, 8 RUTGERS J. OF COMPUTERS, TECH. & L. 173, 173 (1981), for a list of some near-disasters brought about by computer malfunction. See Elson, *Research into Equipment Malfunctions Intensifies*, AVIATION W., Oct. 27, 1980, at 54-55; Aetna Casualty and Surety Co. v. Jeppesen & Co., 642 F.2d 339 (9th Cir. 1981).

4. For an excellent discussion of liability for computer-caused injuries in this field, see Brannigan & Dayhoff, *Liability for Personal Injuries Caused by Defective Medical Computer Programs*, 7 AM. J. L. & MED. 123 (1981).

5. In Arizona State Highway Dep't v. Bechtold, 105 Ariz. 125, 460 P.2d 179 (1969), a faulty "electronic computer-type control" used in the traffic signals at an intersection malfunctioned, causing all the signal lights to be green at the same time. *Id.* at 128, 460 P.2d at 182. Several traffic accidents occurred as a result, and one of the injured motorists sued the highway department. The court held for the motorist based on the negligence of the highway department in not having any form of preventive maintenance of the signals.

For the most part this Comment will be concerned only with physical injuries due to computer program defects. Many purely economic injuries occur daily in the computer field, including loss of information, injury to business, and money spent trying to compensate for computer caused errors.

There is still a great deal of controversy over whether or not purely economic injuries can be compensated under tort law at all. See J'aire Corp. v. Gregory, 24

theories are applied to those cases involving personal injuries as well, it might be difficult for a plaintiff to recover. The plaintiff would have to show the computer programmer failed to use due care in creating or copying the program. This burden might include pointing out the very mistake among the thousands of bits of information in the program and proving that injury is reasonably foreseeable. This would have to be done although the injured person might have no idea how or why a computer operates.[6]

A more sensible solution would be to apply strict products liability principles and eliminate the need for proving negligence. Unfortunately, the problem with applying strict products liability law to computer programs stems from the very nature of the program itself. To apply strict products liability law, there must first be a defective product. However, by today's standards there is a serious question whether a computer program can be considered a product at all.

This Comment will examine the problems involved in applying strict products liability law to computer programs. Two related problems emerge when considering a computer program as a product. The first concerns the intangible nature of the program itself, and whether such an intangible item could ever be considered a product. The second basic problem is whether the transaction surrounding the sale of a computer program involves the transfer of a product or a service. Even if intangible items are covered by strict liability rules, the rules still might not apply to computers if a program is considered to be nothing more than a service provided by the programmer.

To better understand the nature of the problems that arise when computer programs are placed under traditional products li-

---

Cal. 3d 799, 598 P.2d 60, 157 Cal. Rptr. 407 (1979), for a case allowing such recovery. At the present time economic loss in the computer field is being handled pursuant to warranty and contract law, which does not cover all situations, since many computer companies are beginning to write disclaimers and exculpatory clauses into their contracts. *See* Gemignani, *supra* note 3, at 177 n.16.

6. Of course, in a strict product liability cause of action, the plaintiff would still have to show that the program was defective. This is usually an easier task than proving negligence; often the nature of the accident itself will demonstrate the program's defectiveness. If a transport robot is supposed to carry a load of highly toxic gases in special containers across a room, but instead drops them halfway across, there is obviously a defect. The plaintiff might have to show the defect was in the program itself rather than the mechanical part of the machine, but this would probably be easier than isolating individual acts of negligence.

ability law, a brief overview of some basic elements of computer programming is helpful.

THE ELEMENTS OF COMPUTER SOFTWARE

Most people today are probably familiar enough with computers to know that there is more to them than a metal casing and keyboard. For the purpose of this Comment, there are two important basic components of a computer: the hardware (the tangible part of the machine), and the software (the intangible part). According to one definition, the hardware is that part of the computer "you can touch—the integrated circuits components, the wires, switches, lights, keyboard, power supply," and similar things. The software includes programs and data. It has been described as "entities that exist as patterns."[7] In another analysis, computer software is defined as "all those aspects of the computer which are not hardware."[8]

If one focuses on the computer programs[9]—the actual electronic directions and data that cause a computer to perform—there can be vast discrepancies in construction, function, and marketing of various types of programs. Many computer programs are sold in "packages" either with or without accompanying hardware. This type of program often resembles a cassette tape, ready to be placed into a machine and perform. Other programs are specialized for a particular company, and can often be programmed into an existing computer system by the programmer. Still others go through several sets of programmers before they reach the company that uses them.

Often the packages consist of several programs: one to run the basic "life-functions" of the machine, like the part of the human brain that controls the heart and lungs; another set of programs controls other general activities of the computer; still another runs the computational or industrial functions sought by the user.

---

7. R. DIDDAY, HOME COMPUTERS: 2^{10} QUESTIONS AND ANSWERS 1 (1977).

8. Note, *The Revolt Against the Property Tax on Software: An Unnecessary Conflict Growing Out of Unbundling*, 9 SUFFOLK U. L. REV. 118, 121 (1975). This definition would include not only programs and data, but also manuals and perhaps even servicing and maintenance. Obviously if this definition is used, some lines must be drawn between the different elements of "software" before products liability can be applied. Maintenance and upkeep of the computer would probably be considered services rather than products, and thus would not qualify for products liability.

9. A program has also been defined as a "list of instructions" given to a computer so that "it can do a specific job." R. MOODY, THE FIRST BOOK OF MICROCOMPUTERS 11 (1978).

CLASSIFYING THE COMPUTER PROGRAM

Is a Program a Product?

The real problem with applying strict products liability law to computer programs begins with the classification of the computer program or software. For an injured plaintiff to collect under products liability law, he must be injured by a defective product.[10] The question is: can a computer program be considered a product within strict products liability law? If it is a product, strict liability law should apply. However, if it is considered a service performed by the computer programmer, products liability law would likely be inappropriate.[11] Before considering the classification of computers, it is necessary to look at the way current tort law defines a product for purposes of strict products liability.

Section 402A of the *Restatement of Torts* provides the most familiar explanation of the strict products liability theory.[12] The

10. "[T]he term 'products liability' normally contemplates injury or damage caused by a defective product . . . ." W. KIMBLE & R. LESCHER, PRODUCTS LIABILITY 2 (1979); *see also* RESTATEMENT (SECOND) OF TORTS § 402A (1965) (accepted explanation of strict products liability).

11. For the rationale behind denying strict liability to services, see *infra* text accompanying notes 49-60. For a further discussion of the product-service distinction, see W. KIMBLE & R. LESCHER, *supra* note 10, at 99-102; Note, *Strict Liability in Hybrid Cases*, 32 STAN. L. REV. 391 (1980).

12. The RESTATEMENT (SECOND) OF TORTS § 402A (1965) states:

> Special Liability of Seller of Product for Physical Harm to User or Consumer
>
> (1) One who sells any product in a defective condition unreasonably dangerous to the user or consumer or to his property is subject to liability for physical harm thereby caused to the ultimate user or consumer, or to his property, if
>
> (a) the seller is engaged in the business of selling such a product, and
>
> (b) it is expected to and does reach the user or consumer without substantial change in the condition in which it is sold.
>
> (2) The rule stated in Subsection (1) applies although
>
> (a) the seller has exercised all possible care in the preparation and sale of his product, and
>
> (b) the user or consumer has not bought the product from or entered into any contractual relation with the seller.

Section 2-105(1) of the Uniform Commercial Code is slightly more helpful. It defines "goods" as "all things . . . which are moveable at the time of identification to the contract for sale . . . ." U.C.C. § 2-105(1) (1977). This definition does not include money, investment securities, and choses in action. *See generally* Quad County Distrib. Co. v. Burroughs Corp., 68 Ill. App. 3d 163, 24 Ill. 818, 385 N.E.2d 1108 (1979) (assumption that programs come under the U.C.C. provisions for sales of goods). For the purposes of contract law, at least, a computer program can be considered a product because it can be moved and transferred to any other party at the time of the sale.

law of products liability theory as discussed in section 402A, however, is more concerned with the definition of defectiveness than an examination of what constitutes a product.

Though no consistent definition of a product has been developed, certain generalizations can be made. "Products" tend to be manufactured items like autos or soda bottles,[13] produced by companies in the chain of production and marketing of such goods.[14] Services are functions performed by people like architects, engineers, and doctors.[15] The classification of a transaction as either the sale of a product or a service is usually fairly straightforward.

The distinction between products and services is not so easily made, however, when dealing with computer programs and programmers. A computer program possesses many of the characteristics of both a product and a service. It can be manufactured on a mass scale,[16] but it can also be fed individually into a computer by a single programmer. The fact that programs can be individualized alone is not conclusive. Many items that are usually considered manufactured goods (such as automobiles) can be handcrafted to suit an individual customer.[17]

A more serious problem appears when the individual programming capability is added to the intangible nature of the computer program. The nature of programs caused at least one noted author to determine that a computer program should not be defined as a product.[18] According to his theory, a computer program is merely a process, no more a product than an industrial process.[19]

The Computer Program and Intangible Items

There is no absolute rule that restricts the definition of products to tangible items. The *Restatement* makes no mention of

13. Exploding soda bottles have been considered eligible for strict liability since the earliest products liability cases. *See* Escola v. Coca Cola Bottling Co., 24 Cal. 2d 453, 150 P.2d 436 (1944).

14. Ursin, *Strict Liability for Defective Business Premises—One Step Beyond* Rowland *and* Greenman, 22 U.C.L.A. L. REV. 820 (1975).

15. Performers of services are generally not liable under strict liability. *See* Queensbury Union Free School Dist. v. Jim Walker Corp., 91 Misc. 2d 804, 398 N.Y.S.2d 832 (1977); Hoven v. Kelbe, 79 Wis. 2d 444, 256 N.W.2d 379 (1977).

16. For instance, Apple computers sells a line of mass-produced software for home computers.

17. In some cases strict liability has not been applied when the sellers are private individuals who do not operate commercially. It is possible that strict liability law might not be applicable in any case to a truly one-of-a-kind item. Bacher v. Pearson, 479 P.2d 319, 328 (Alaska 1970); Price v. Shell Oil Co., 2 Cal. 3d 245, 254, 466 P.2d 722, 728, 85 Cal. Rptr. 178, 184 (1970).

18. Freed, *Products Liability in the Computer Age*, 12 FORUM 461, 478 (1977).

19. *Id.* at 473-77.

tangibles at all, though section 402A does come under the heading labelled "Suppliers of Chattels."[20] There is a strong tendency under modern law, however, to assume that "chattel" refers only to tangible items.[21]

In tax law the courts have dealt with the issue of computer program intangibility for several years. The results have been mixed. Several courts have ruled that software is not taxable as personal property because of the intangible nature of the program.[22] Others have held that the tangible output is taxable even though the program itself may not be.[23] One court held that, even considering the intangible nature of the program, it should be taxable for its full value though the materials it is recorded on are worth far less.[24]

Two problems would arise in tort law if tangible output, such as graphs and printouts, was considered the product in place of the program itself. First, because the "product" would be the output, it would be necessary to show that the output itself was defective. This might be difficult if the defect which caused the injury was not apparent in the printed output.[25] Second, basing products liability on tangible output is at best a temporary solution. Today many programs function with no tangible output whatsoever. If programs are to be judged as products, it must be on a more basic level than this.

In the field of sales tax, the fluctuation in results has been similar, once again due to the intangible nature of the program.[26]

20. This was pointed out in Note, *Negligence: Liability for Defective Software*, 33 OKLA. L. REV. 848, 849 (1980). The Note observed that many courts have extended the definition of products beyond the restrictive idea of "chattels." *Id.* at 849 nn. 9-13.

21. Note, *supra* note 20, at 848-49. *See generally* Freed, *supra* note 18 (discussion of why intangibles such as computer programs should not be considered products).

22. *See, e.g.*, Dist. of Columbia v. Universal Computer Associates, Inc., 465 F.2d 615 (D.C. Cir. 1972); *see* Bryant & Mather, *Property Taxation of Computer Software*, 18 N.Y.L.F. 59 (1972).

23. Accountants Computer Serv., Inc. v. Kosydar, 35 Ohio St. 2d 120, 298 N.E.2d 519 (1973).

24. Greyhound Computer Corp. v. State Dep't of Assessments and Taxation, 271 Md. 674, 320 A.2d 52 (1974).

25. It has been said that the output is merely the computer's way of communicating the calculations it has made, in much the same way humans use speech to communicate the decisions of the human mind. Therefore, output could no more be considered tangible than any written expression of an intangible idea could be. Freed, *supra* note 18, at 474-77.

26. *See, e.g.*, Bullock v. Statistical Tabulating Corp., 549 S.W.2d 166 (Tex. 1977);

However, recently some states have passed laws making certain types of computer programs taxable.[27]

In any case, the definition of a product for taxation is not the same for torts. Nevertheless, it is significant that in many taxation cases the courts recognize that computer programs do have a valid existence beyond the programmers who make them. Whether the courts consider the essence of the program to be the series of electronic impulses or the tangible printout, judicial decisions indicate that there is something being exchanged in a computer transaction beyond a mere service.

Courts must also address the intangibility of computer programs in patent law litigation. Computer companies have been plagued with program "piracy," that is, companies without authorization duplicating and, in effect, stealing programs.[28] In the past the courts refused to allow many forms of patenting of computer programs.[29] This was especially true in cases where the program did not produce a "physical result."[30] The courts have begun to relax the doctrines to allow some program patents. The Supreme Court recently held that a program was patentable as part of a trade process for curing rubber.[31] Although not necessarily a complete affirmation by the courts that programs are patentable, this decision is another indication that courts are finding the barrier of intangibility less important.

Other Intangibles Under the Law of Products Liability

If one assumes that a computer program is nothing more than a series of instructions coded in intangible electronic impulses, it will be helpful to examine the courts' treatment of other intangibles under products liability law. An area that has received

Citizens Fin. Corp. v. Kosydar, 43 Ohio St. 2d 148, 72 Ohio Op. 2d 83, 331 N.E.2d 435 (1975).

27. California has passed laws on the subject. Under California law, however, most software is not taxable unless the company being taxed fails to differentiate between hardware and software in the total price value of the computer. If the two are itemized, only the hardware and "basic operational programs" are taxable. CAL. ADMIN. CODE tit. 18, §152 (1981). Also under California tax law, the "storage media," (tapes, cards, and printouts) are taxable. CAL. REV. & TAX. CODE §§ 995, 995.1 (West Supp. 1981).

28. For a discussion of the copyright and pirating problems, see Stern, *Another Look at Copyright Protection of Software: Did the 1980 Act Do Anything For Object Code?*, 3 COMPUTER L.J. 1 (1981). According to sources within the computer industry, some of the "pirating" is even international. 3 MIS WEEK, June 30, 1982, at 1, col. 1.

29. *See* Moskowitz, *The Patentability of Software Related Inventions After* Diehr *and* Bradley, 63 J. PAT. OFF. SOC'Y 222 (1981).

30. *See, e.g.*, Gottschalk v. Benson, 409 U.S. 63 (1972); *see also* Diamond v. Diehr, 450 U.S. 175, 193 (1981) (Stevens, J., dissenting).

31. Diamond v. Diehr, 450 U.S. 175 (1981).

much attention recently is strict products liability for injury caused by electricity. Some courts have allowed recovery if homes were damaged by "defective" electric current.[32] Several other courts have denied recovery on the grounds that the electricity was never sold,[33] that it was never transferred to the consumer,[34] or that it was not defective.[35] Most of these courts, if they addressed the idea of the intangible product at all, assumed that electricity was a product.[36]

Electricity has no tangible form beyond its effect. It is a form of energy, similar to the impulses of a computer program. If courts are willing to extend strict products liability to cover defective electricity, it seems reasonable that they extend the doctrine to computer programs.

The Product-Service Distinction

If the issue is resolved such that a computer program has the capability of being a product—that either intangible items can be products or that the program itself is tangible enough to be one—a more difficult problem arises. This concerns a computer program's classification as either the sale of a good or performance of a service. Just because a computer program is tangible does not always mean that the major transaction involved in its sale is the purchase of a product.[37]

The Policy Arguments

Important policy reasons have led courts to adopt strict liability for products and not services. Several of these policies could also apply to computer programs.[38] These policies vindicate goals such as loss spreading, accident reduction, and victim

32. Ransome v. Wisconsin Elec. Power Co., 87 Wis. 2d 605, 275 N.W.2d 641 (1979).

33. *See, e.g.*, Genaust v. Illinois Power Co., 62 Ill. 2d 456, 343 N.E.2d 465 (1976).

34. Petroski v. Northern Indiana Public Service Co., 171 Ind. App. 14, 30-31, 354 N.E.2d 736, 747 (1976).

35. Erwin v. Guadalupe Valley Elec. Co-op, 505 S.W.2d 353 (Tex. 1974).

36. Ransome v. Wisconsin Elec. Power Co., 87 Wis. 2d 605, 275 N.W.2d 641 (1979); Petroski v. Northern Indiana Pub. Serv. Co., 171 Ind. App. 14, 354 N.E.2d 736 (1976).

37. The courts have been very reluctant to extend strict liability when, although there is a product involved, the actual thing being sought by the purchaser is a service. *See* W. KIMBLE & B. LESCHER, *supra* note 10, at 99-100.

38. For a good discussion of some of the policy arguments involved, see Note, *supra* note 20.

compensation.[39]

Products liability looks at the relative positions of the victim and the defendant. Often the physically injured victim is an average person with little or no background in the field involved in the litigation.[40] It might be very difficult for him to identify the specific act of negligence responsible for the defect that ultimately caused his injury—especially in an industrial process he does not understand.[41] The manufacturer is in a much better position to identify the problem both before and after the accident.[42]

The accident reduction goal applies to the computer industry. The average person has little idea how a computer functions or why a program works. The programmer, like other manufacturers, is in a better position than the injured person to recognize the mistake and correct it. The programmer can also test the program before he places it on the market. Even if the injured plaintiff has some knowledge of computers, it may be difficult for him to isolate the specific act of negligence that led to injury. This is especially true in a field such as computer programming where mistakes are an everyday occurrence.[43]

Because many computer companies today produce software in mass market fashion, the goal of loss compensation also applies. The computer industry can better absorb the cost of injuries,

---

39. Justice Traynor discusses these policies in his concurring opinion in Escola v. Coca Cola Bottling Co., 24 Cal. 2d 453, 150 P.2d 436 (1944) (Traynor, J., concurring). Loss spreading involves putting the financial burden on the manufacturer rather than the injured person, because the manufacturer can then "spread the loss" among his customers through higher rates. All the people who use the product, in effect, become insurers for it. *Id.* at 462, 150 P.2d at 441.

Accident reduction refers to the manufacturer's ability to make the product relatively safe, or at least tested, before he places it on the market. If the manufacturer is held responsible for all injuries caused by the product's defectiveness, he will have the incentive to try as much as is economically feasible to make the produce safe. *Id.* at 462, 150 P.2d at 440.

Victim compensation is just that—a way of providing monetary relief to an injured victim who is probably in a poor financial position due to his injury. It is felt that, as between the innocent victim and the manufacturer who placed the defective product on the market, the manufacturer should be the one to pay the price of the injury. *Id.*, 150 P.2d at 441.

40. "An injured person, however, is not ordinarily in a position to refute such evidence or identify the cause of the defect, for he can hardly be familiar with the manufacturing process as the manufacturer himself is." *Id.* at 463, 150 P.2d at 441.

41. *Id.* at 462-64, 150 P.2d at 441-43.

42. "It is evident that the manufacturer can anticipate some hazards and guard against the recurrence of others, as the public cannot." *Id.* at 462, 150 P.2d at 441-42.

43. For examples of common errors, see *Those Computers Get Temperamental*, U.S. NEWS AND WORLD REPORT, Aug. 20, 1979, at 54-55. For an excellent discussion of the reasons behind program failures, see Gemignani, *supra* note 3, at 181-84.

either through insurance[44] or by adjusting prices, than can the injured victim. This is true even if the programming operation is small.[45]

Two other important policy reasons argue for strict products liability in this field. The first concerns a strange problem, unique to computer programs, that is not completely understood. Some scientists believe computer programs can occasionally malfunction spontaneously without any evidence of negligence.[46] If a program became defective, even temporarily under this circumstance, the victim would be unable to recover under negligence. This situation demonstrates a strength of strict products liability. In *Escola v. Coca Cola Bottling Co.*, Justice Traynor noted: "Even if there is no negligence . . . public policy demands that responsibility be fixed wherever it will most effectively reduce the hazards to life and health inherent in defective products that reach the market."[47]

The second reason to adopt strict liability for computer programs is to prevent random application of the law. Increasingly, computers are being used in devices like automobiles. It would be undesirable to permit an injured consumer to collect under strict products liability for a defective steering mechanism, but not for a defective computer program in the car which may have caused the same injuries.[48] Yet if computer programs are treated only as services, this situation could result.

The Distinctions Between Programming and Services

Another way to examine the product-service distinction is to

44. According to Freed, *supra* note 18, at 477, however, many computer companies are not buying insurance.

45. Despite the size of the operation, it is still preferable to hold the manufacturer responsible for the defective item he puts on the market, rather than place the burden on the innocent victim. *See* Escola v. Coca Cola Bottling Co., 24 Cal. 2d at 462, 150 P.2d at 441.

46. *Bad Bits*, SCI. AM., Feb. 1980, at 70. If these spontaneous defects arose after the program was sold, there might be a problem applying strict liability, since the defect must exist at the time the product leaves the hands of the manufacturer. RESTATEMENT (SECOND) OF TORTS § 402A (1965). However, since it is not fully understood why these spontaneous defects occur, an argument could be made that they are due to some latent defect in the original program that manifested itself later.

47. Escola v. Coca Cola Bottling Co., 24 Cal. 2d at 462, 150 P.2d at 440 (Traynor, J., concurring).

48. For a case dealing with a defective car computer, see Volvo of America Corp. v. Wells, 551 S.W.2d 826 (Ky. Ct. App. 1977).

compare the activities of a programmer with those professions currently protected by the "service" label. "Services" in the context of strict products liability tend to be provided by smaller businesses without the benefit of the control of assembly-line production. The true service transaction is geared to a particular circumstance; it can rarely be duplicated. Repairpersons, for instance, are faced with a new product and circumstance every time they operate, so there is little chance for quality control or defect testing.[49] Although some computer programming companies fit into this small, private business idea, many closely resemble large manufacturers.[50] Software, unless it is custom-made, is often created with a particular type of machine in mind, with ample opportunity for testing.[51]

Strict products liability is not usually a deterrent to the hazards of a true service, because the nature of the task is dictated by the situation. In the computer field, however, the goal of accident reduction would be served by applying strict products liability. The race to market new programs in this fast-growing field occasionally causes goods to be placed on the market before being completely tested.[52] Strict products liability provides incentive to companies to improve testing and planning of new programs.[53]

Often the sales-service distinction is employed to protect professionals, especially doctors.[54] Because each patient has different problems, a doctor's performance can rarely be "mechanical or routine."[55] Strict liability would not help prevent risks that negligence law does not currently cover. In fact there is some belief that holding doctors and professionals strictly liable could actually reduce the quality of medical care.[56] Instead of spreading the

---

49. Note, *supra* note 11, at 397-98.

50. A few of the better known large corporations are Apple, International Business Machines (IBM), Burroughs, National Cash Register (NCR), and Honeywell.

51. Examples of programs made for a particular type of machine are those being put into cars. *See* Volvo of America Corp. v. Wells, 551 S.W. 826 (Ky. Ct. App. 1977). Other examples include computer systems that control building interior environments. Honeywell Corporation alone has sold more than 1,500 of these since 1975. *See Those Computers Get Temperamental*, U.S. NEWS & WORLD REPORT, Aug. 20, 1979, at 54. A more specific example is the "laboratory package" put out by the Medical Data Corporation, which automatically processes laboratory results.

52. *Snafus That Delay New Products*, BUS. WK., June 1, 1981, at 110.

53. "Moreover, holding manufacturers strictly liable forces them to balance the cost of maintaining product quality standards against the burden of recompensing all injuries from substandard goods, and therefore encourages an efficient level of investment in product safety." Note, *supra* note 11, at 394.

54. *See* Clark v. Gibbons, 66 Cal. 2d 399, 426 P.2d 525, 58 Cal. Rptr. 125 (1967) (Traynor, J., concurring), in which Justice Traynor used the same rationale to object to the court's adoption of an expanded res ipsa loquitur doctrine for doctors.

55. Newmark v. Gimbel's Inc., 54 N.J. 585, 596, 258 A.2d 697, 703 (1969).

56. *See generally* King, *In Search of a Standard of Care for the Medical Profes-*

loss among a large number of consumers, a doctor would only put the financial burden back on his own private patients. Some of these patients might eventually be unable to afford medical assistance at all if rates continue to rise.[57]

The computer industry generally does not suffer from these problems. In most cases the computer industry deals with a larger and more affluent clientele than do doctors. This clientele can afford to pay if the programming company is forced to raise prices to cover losses. Testing for mistakes before the program is delivered is possible, and correction of those errors both before and after delivery is practical. Computer companies today often employ "debuggers," specialists who work to eliminate the "bugs" or defects in a program after it has been delivered and is in use.[58]

Programmers, unlike doctors or other professionals, are generally employed by large companies or corporations. Programmers do not therefore risk loss of reputation to the extent individual doctors would if held strictly liable.[59] In fact, strict products liability may actually eliminate the stigma of negligence programmers might otherwise suffer.[60]

Programs as a Product or Service in Other Areas of the Law

Despite different mechanical and computerized methods developed in an attempt to prevent piracy, program theft continues to remain a serious problem. That computer programs can be stolen and duplicated demonstrates their product nature. It would be very difficult to steal a service.

Computer programs have been compared to industrial secrets,

---

sion: The "Accepted Practice" Formula, 28 VAND. L. REV. 1213, 1227-29 (1975) (discussion of the problems caused by medical malpractice).

57. Justice Traynor felt that an expansion of doctors' liability could result in an "undesirable limitation on the use of procedures involving inherent risks of injury even when due care is used." Clark v. Gibbons, 66 Cal. 2d 399, 424, 426 P.2d 525, 542, 58 Cal. Rptr. 125, 142 (1967) (Traynor, J., concurring).

58. Freed, *supra* note 18, at 466-67.

59. Computer programmers are, however, concerned with the threat of possible liability. *See* Nycum, *Liability For Malfunction of a Computer Program*, 7 RUTGERS J. COMPUTERS, TECH., & L. 1, 22 n.90 (1979).

60. *Cf.* Clark v. Gibbons, 66 Cal. 2d at 421, 426 P.2d at 540, 58 Cal. Rptr. at 140. Negligence law concentrates on the actions of individuals; it brands their conduct as good or bad. Strict products liability, on the other hand, examines only the product itself, and whether it was defective or unreasonably dangerous. The conduct of the programmer within the company is irrelevant. Strict liability used in this way could actually protect individual reputations, rather than harm them.

with the printouts resembling technical manuals.[61] In this analysis the "secrets" would be possible targets for theft and still not be "goods." However, a trade secret is not usually a marketed item, mass-produced for the public. Trade secrets are used to create products which are then marketed. If part of the secret process causes a product defect the manufacturer will still be held strictly liable, notwithstanding that a secret industrial process created the device.[62] In the computer industry it is the program that is sold and bought. The program is not merely a trade process for developing hardware.[63]

A computer program could also be compared to a car instead of a trade secret. The ultimate purpose of the car is transportation, but the transaction involved is for the purchase of the item itself, not the transportation. Similarly, though the ultimate function of the computer program is to provide those activities a company or

---

61. Freed, *supra* note 18, at 467-68.

62. The type of defect in this case, and in most computer program cases, is a design defect. *See* Nycum, *supra* note 59, at 15-17. Types of defects can be separated into two categories: manufacturing defects, in which a product is accidently made differently from the other products like it (for instance, the defective cola bottle which explodes when touched); and design defects, in which a product "may be exactly the way the manufacturer intended that it be made," and still be defective. Note, *Strict Liability in Tort: Is It Applicable to Design Defects?*, 20 WASHBURN L. J. 600, 601 n.8 (1981). Computer programs can have both kinds of defects (a manufacturing defect might occur, for instance, if something went wrong with one particular program while it was being copied for mass production, but all others like it were correct), but usually a defective computer program will be said to have a design defect. This is because most of the bugs or problems in a program occur as it is being initially created and punched into a computer. If a defective program is then copied, all the copies will contain that same error—they have been defectively designed.

The California court in Barker v. Lull Eng'g Co., Inc., 20 Cal. 3d 413, 426-27, 573 P.2d 443, 454-56, 143 Cal. Rptr. 225, 236-38 (1978), developed a two-alternative test for determining if a product is defectively designed: "if (1) the plaintiff proves that the product failed to perform as safely as an ordinary consumer would expect when used in an intended or reasonably foreseeable manner, or (2) if the plaintiff proves that the product's design proximately caused injury and the defendant fails to prove, in light of the relevant factors, that on balance the benefits of the challenged design outweigh the risks of danger inherent in such design."

With a defective computer program in California, the plaintiff would have a choice of either theory, but he would probably be more successful arguing under the first as a general rule, since a computer that kills or injures someone is certainly not living up to the public's expectation of it.

63. A distinction could be made between two separate types of services involved in the computer industry:

(1) the service the programmer performs in writing the program for the machine, and
(2) the services performed by the machinery itself (accounting, welding).

Although it is not clear that a distinction has been made between these two types of "services," for the purposes of this Comment, service will usually refer to (1). The discussion comparing computers to cars and transportation is the one major exception.

consumer desires—running a welding robot or keeping accounting records—it is the program itself that is usually bargained for and bought, not the accounting service.

Closely related to the ideas of patenting and piracy is the typical manner in which computer programs are marketed. Many programs are now being mass-marketed. Apple computers, for instance, are sold from a chain of retail outlets advertising similar, if not identical, lines of computer programs.

Some people have already organized computer clubs, much like book clubs or cosmetic clubs, with lower prices on software for members.[64] At a local seminar for teachers on the use of computer teaching aids, teachers who were discussing programming their own machines were told that it was impractical because so many good teaching programs already exist, ready to be bought. Apparently, even if the courts are hesitant, both industry and public opinion now look on computer programs as a commodity to be bought and sold like a cassette tape or a hair dryer.[65]

Newmark, Computer Programs, and the Sales-Service Hybrid Cases

Probably the most important area in which the courts have chipped away at the traditional rigid definitions of the word "product" is in the so-called "hybrid" cases. These involve a transaction that is both a sale of a product and a service. Because a computer program may involve the sale of a product (the completed program itself), *and* a service provided by the programmer in creating the program, another approach is to consider the sale of a program *both* a product and a service.

The forerunner of the hybrid cases is *Newmark v. Gimbel's Inc.*[66] In this case a beauty salon applied defective conditioner to a customer's hair, injuring the customer. The court held against the beauty salon under strict products liability, because the transaction involved was actually a "hybrid" of the sales-service dis-

64. One recently advertised club is the American Software Club, Inc., Millwood, N.Y. 10546.

65. Another important point emphasized in the product area is the ownership of computer programs. While services cannot be owned, programs can be owned like any other product. A program is manufactured, often wholesaled or retailed, and passed along to the consumer or industry. This ownership distinction was noted in Brannigan & Dayhoff, *supra* note 4 at 131-32.

66. 54 N.J. 585, 258 A.2d 697 (1969).

tinction; it was a combination of the two. The customer was purchasing the defective conditioner along with the treatment. Due to the hybrid nature of the transaction, the customer could actually recover for the injury caused by the defective product even though the beauty salon was supplying a service. The "hybrid theory," as developed in *Newmark* and later cases, consists of two general steps. First, it must be determined that there was both a product and a service involved. Second, the courts look at the defect itself to see if it arose out of the product part or the service part of the transaction.[67]

Later courts have expanded the doctrine to include blood transfusions and situations where the manufacturer improperly installed his own product.[68] Some of the courts that ruled for electricity as a product used much of the sales-service hybrid reasoning.[69]

Under present interpretations, the hybrid theory probably would not cover most of the computer program cases. This is due to the difficulty in determining in which part of the transaction—the product or service portion—the defect arose. If the completed program was defective when sold and caused injury, the defect would seem to arise out of the product end of the transaction. However, that same program could very easily be defective because of an error made by the computer programmer when creating the program. If the creation of the program is considered a service, then the error might have arisen out of the service part of the transaction.[70] Under this interpretation, the service used to create the product and the product it creates are so closely related that it is difficult to separate the two. Yet it is inadequate to

67. *See generally* Note, *supra* note 11 (discussion of different ways of looking at this two-step process).

68. Hoffman v. Misericordia Hosp. of Philadelphia, 439 Pa. 501, 267 A.2d 867 (1970); Johnson v. Sears Roebuck & Co., 355 F. Supp. 1065 (E. D. Wis. 1973); Paint Prods. Co. v. AA-1 Steel Equip. Co., 35 Conn. Supp. 52, 393 A.2d 1317 (1977).

69. *See, e.g.*, Buckeye Union Fire Ins. Co. v. Detroit Edison Co., 38 Mich. App. 325, 196 N.W.2d 316 (1972).

70. A much less complicated problem would occur if the error was caused, not during the original creation of the program on paper, but during the time the newly-created program was entered into a machine. Many program errors are caused by a programmer hitting the wrong keys while entering a program into the computer. If the initial program was designed correctly, but punched into the machine incorrectly, the programming service itself would not be defective. If it is argued that the act of putting the program into the machine is considered part of the service, products liability could still apply under the holding of Paint Prods. Co. v. AA-1 Steel Equip. Co., 35 Conn. Supp. 52, 393 A.2d 1317 (1977), in which a manufacturer who defectively installed his own goods was held liable for the consequences of that improper installation under strict products liability.

If the same company that designed the program improperly entered it into the computer, the situation would resemble either a manufacturer who defectively installed his own goods, or an item which was simply manufactured incorrectly (if

merely label the transaction a service because of this difficulty. As suggested earlier, a program involves something more concrete than just a service.

One solution would be to expand the *Newmark* doctrine to cover cases where computer software causes injury. Both *Newmark* and computer program cases involve a product sold to the customer that injures him. The companies that do the programming are businesses like the beauty parlor, not professionals like doctors or dentists who have an "intimate relationship to public health and welfare."[71] The same policy reasons that originally led the courts to expand strict liability into the hybrid area also apply to computer programs. In both situations there is "partly the rendering of a service, and partly the supplying of goods for a consideration."[72]

This approach would relieve the injured plaintiff from having to determine whether the defect arose when the program was designed, when the newly-created program was first punched into the machine, or because the program was copied incorrectly before it was sold. If this distinction is necessary, it would place an additional burden on the plaintiff, and create confusion over whether negligence or strict liability should apply on a case-by-case basis. An expansion of *Newmark* would simplify matters and facilitate compensation of injured victims.[73]

---

the process of entering the program is considered part of the manufacture of the item). *See* Gemignani, *supra* note 3, at 187.

At this point it becomes important to draw the line between the work that goes into producing the actual program, and the maintenance that goes on afterward. Any maintenance that occurs after the program has been delivered would almost always be considered a service. If the defect occurred because of negligence in the servicing, it is unlikely that the injured party could get a judgment under strict products liability. A manufacturer of a washing machine, for example, should not be held strictly liable for the negligence of a repairperson. If, on the other hand, the defect existed before the unit was repaired, strict liability would probably apply.

71. Newmark v. Gimbel's Inc., 54 N.J. 585, 590, 258 A.2d 697, 703 (1969).

72. *Id.* at 588, 258 A.2d at 701.

73. Another problem in the products liability field arises with programs that are altered either by the manufacturer or by the company that purchases them. Some programs, in fact, are specifically designed to be altered by the company that will be using them.

A related situation occurs when several different groups of programmers create different parts of the programs necessary to run a particular piece of hardware. The complete program package could actually consist of three (or more) different programs: one to perform "life functions"; one for the computer's general activities; and a third to perform the actual activity for which the program or machine

CONCLUSION

With the increasing use of computer programs in industry, home, and commercial work, it is likely there will be physical injuries to people from defective computer programs. The most practical way to meet this situation is to apply the principles and policies of strict liability for defective products to computer-caused injuries. The obstacle to this application is the nature of the computer program itself—it is not a tangible chattel in the traditional sense, and has some elements of a service. However, all the policy reasons that led to the adoption of strict products liability apply to the computer industry. Also, when one compares the courts' reactions to other intangible items (such as electricity) that cause injury, and the reception of computer programs in other areas of the law, one finds a relaxation by the courts of the traditional rigid definition of "products." At the very least, a program could be thought of as a hybrid transaction involving the sale of both a product and a service. This would bring the computer program under an expanded version of the *Newmark* doctrine and allow recovery based on the dual nature of the program.

SUSAN LANOUE

---

was purchased. If the machine malfunctions and injures someone due to program error, it may be difficult for the consumer to show which company was responsible for the defect.

Most of these problems can be handled under existing products liability law. In the case of the subsequent alterations by the purchaser, it would be necessary to determine if the defect existed before the alterations were made. W. KIMBLE & R. LESCHER, *supra* note 10, at 90-91. If so, strict products liability would probably apply. If the defect was brought about in the alterations, the plaintiff would probably have to show negligence. *Id.* at 91-92.

With the multiple programmers problem, again current products liability law can probably come to the rescue. Many goods put on the market are made up of components from different manufacturers added together to make a whole. Under products liability law a component parts manufacturer can be held liable if "there [is] . . . some evidence, or at least a contention, that one of the component parts malfunctioned in some way." *Id.* at 59-60.