LIABILITY ISSUES WITH ARTIFICIAL INTELLIGENCE SOFTWARE

I. INTRODUCTION

"Computers can only issue mandatory instructions—they are not programmed to exercise discretion."[1]

It has been fifteen years since this statement was made by a New York court,[2] and it is no longer strictly true. What only decades ago was the stuff of science fiction is today the reality of science: computers capable of solving problems by replicating human thought processes.[3] Through the use of artificial intelligence (hereinafter AI),[4] programs are available that provide tax advice,[5] diagnose medical conditions,[6] and configure computers.[7] With continued research, the scope of AI programs will broaden,[8] and as it penetrates markets for critical services and processes it is likely the potential for catastrophe will increase.[9] When one considers that conventional software

---

1. Pompeii Estates, Inc. v. Consolidated Edison Co., 397 N.Y.S.2d 577, 580 (N.Y. Civ. Ct. 1977).

2. *Id.*

3. Cariad Hayes, *Artificial Intelligence: The Future's Getting Closer*, AM. LAW., Nov. 1988, at 115. Hayes points out that certain "software programs . . . in their complexity . . . imitate certain processes of the human brain." *Id.*

4. Artificial intelligence is a general term used to describe that aspect of computer science "concerned with understanding the nature of intelligent action and constructing computer systems capable of such action." Allen Newell, *Artificial Intelligence*, *in* 2 MCGRAW-HILL ENCYCLOPEDIA OF SCIENCE AND TECHNOLOGY 120 (Sybil P. Parker ed., 7th ed. 1992). "A machine has artificial intelligence when there is no discernible difference between the conversation generated by the machine and that of an intelligent person." ALAN FREEDMAN, THE COMPUTER GLOSSARY: THE COMPLETE ILLUSTRATED DESK REFERENCE 12 (5th ed. 1991) (quoting Alan Turing).

5. "AskDan," a program from Legal Knowledge Systems Inc., does tax calculations and identifies tax loopholes. Lance B. Eliot, *Mass Market Applications: They're Here*, AI EXPERT, Dec. 1989, at 9.

6. MYCIN, developed at Stanford University, is used by physicians to diagnose bacterial blood infections. Yi-Tzuu Chien & Jay Liebowitz, *Artificial Intelligence*, *in* 2 ENCYCLOPEDIA OF PHYSICAL SCIENCE AND TECHNOLOGY 1, 14 (Robert A. Meyers ed., 1987). FamilyCare Software from Lundin Laboratories, Inc. gives pediatric advice. Eliot, *supra* note 5, at 9.

7. XCON was developed by Carnegie-Mellon University for use in configuring Digital Equipment Corporation's VAX computers. Chien & Liebowitz, *supra* note 6, at 15.

8. Laurence H. Reece III, *Defective Expert Systems Raise Personal Injury Liability Issues*, NAT'L L.J., Oct. 12, 1987, at 24.

9. It has been reported that a patient died from excess radiation when a computer-

programs have resulted in near misses between commercial jets, the closing of nuclear power plants, and a missile alert triggered by a false indication of a world war,[10] it is easy to predict additional problems could be wrought by a program mimicking human thought. Furthermore, as systems become more complex, malfunction is inevitable.[11] When such failure occurs, who bears the liability? The answer is unclear.

The impact of an AI program error has not yet been considered by the courts and no case law exists providing guidance to the software developer, vendor, or user with respect to the potential liability of each. In addition, there are no statutes dealing with this issue. This comment proposes a standard for liability that eliminates this uncertainty.

This comment first discusses the differences between conventional and AI software,[12] including the special subset of AI known as expert systems,[13] then presents various theories of liability and how they relate to AI.[14] Finally, a proposal that a strict liability standard be applied to AI will be presented.[15]

II. BACKGROUND

A. *Software: Conventional and Artificial Intelligence Systems*

Software is the set of instructions specifying the required steps for data processing by a computer.[16] The term generally includes the system software (i.e. the operating system), application software, and

---

controlled radiation-therapy machine malfunctioned. In another incident, payment was stopped on checks which, due to a software error, paid bondholders $4 million in excess interest. In another instance, twenty sailors were killed due to a problem with a computer-controlled air-defense missile system that malfunctioned as a result of an error. Bob Davis, *Costly Bugs*, WALL ST. J., Jan. 28, 1987, at A1; L. Nancy Birnbaum, *Strict Products Liability and Computer Software*, 8 COMPUTER/L.J. 135, 144 n.64 (1988). *See also* Bev Littlewood & Lorenzo Strigini, *The Risks of Software*, SCI. AM., Nov. 1992, at 62, 62-63.

10. Michael C. Gemignani, *Product Liability and Software*, 8 RUTGERS J. COMPUTERS, TECH. & L. 173 (1981).

11. "No complex computer program has ever been marketed that did not have some defect, somewhere." Hayes, *supra* note 3, at 115 (quoting Robert Cesari). *But see* Littlewood & Strigini, *supra* note 9, at 62 (stating that, in theory, design faults in a program could be detected and removed).

12. *See infra* text accompanying notes 16-75.

13. *See infra* text accompanying notes 35-62.

14. *See infra* text accompanying notes 76-153.

15. *See infra* text accompanying notes 259-61.

16. JOHN T. SOMA, COMPUTER TECHNOLOGY AND THE LAW § 1.06, at 8 (1983).

documentation.[17] Application software is also referred to as a program, "a set of statements or instructions to be used directly or indirectly in a computer in order to bring about a certain result."[18]

1. *Conventional Software*

Conventional programs operate in a linear fashion, manipulating input provided by the user through a specific process to reach a designated output.[19] The process is based on a specific algorithm[20] that produces generally consistent results for given input, with little if any interaction between the user and the program.[21] Such programs provide the basis for familiar computer-based activities in which sequential procedures clearly define actions to solve a problem: Handling large quantities of numbers and symbols for spreadsheets, databases, word-processing, and financial analysis.[22] The programs are generally designed to be mass-marketed rather than custom-designed.[23] There is little, if any, interaction between the end-user of the software and the developer, and the skill of the individual user or specific intended use for the program is not considered prior to sale by the vendor.[24] Despite extensive testing, it is unlikely any conventional program is error-free.[25] Errors or "bugs"[26] can be introduced at a number of points during the development, loading, or operation of the program.[27] The question is not whether there is some risk, but rather what risk level is acceptable to maximize usefulness and minimize liability.[28]

---

17. *Id.*

18. 17 U.S.C. § 101 (1988).

19. G. Steven Tuthill, *Legal Liabilities and Expert Systems*, AI EXPERT, Mar. 1991, at 44, 48.

20. An algorithm is "a well-defined procedure for solving a problem or doing some task." MICHAEL C. GEMIGNANI, COMPUTER LAW app. at 655 (1985).

21. Tuthill, *supra* note 19, at 48.

22. *Id.*

23. SOMA, *supra* note 11, § 3.12, at 89-91.

24. Mass-marketed programs are available indiscriminately on payment of a fee through a variety of channels: Mail-order, retail, or wholesale outlets. RAYMOND T. NIMMER, THE LAW OF COMPUTER TECHNOLOGY ¶ 5.16, at 5-59 (1985).

25. "It has frequently been said that the only error-free program is the one that will never be run again." Gemignani, *supra* note 10, at 185 (quoting A. Pietrasanta, *quoted in* PROGRAM TEST METHODS 1 (W. Hetzel ed. 1973)); *see also* Hayes, *supra* note 3, at 115.

26. A bug is "an error in a program." GEMIGNANI, *supra* note 20, app. at 657.

27. Gemignani, *supra* note 10, at 183-84.

28. *Id.* at 187.

2. *Artificial Intelligence/Expert Systems*

Artificial intelligence programs utilize the knowledge of the relationship between objects and events in a particular focused problem area (the "domain")[29] as the basis for problem solving.[30] Rather than using a mathematical algorithm to manipulate data, AI depends on the "symbolic manipulation of information" through the use of heuristics.[31] Unlike mathematical algorithms, heuristics do not always "work" to give a precise answer; they merely offer a "clue" to the solution.[32] It is by combining all useful ideas or clues and having adequate knowledge about the problem domain that the solution is obtained.[33] Therefore, sufficient knowledge, provided by human experts, is required.[34]

A specific subset of AI is an expert system,[35] a computer program that "emulates the behavior of human experts within a specific domain of knowledge."[36] Unlike human experts, expert systems are "permanent, easy to transfer, easy to document, consistent and affordable."[37] Shortcomings of such programs are that "[t]hey are less creative and less adaptive"[38] than people and are incapable of applying common-sense knowledge to a problem.[39] Because even an expert is not always 100% accurate, the expert system is designed to accommodate uncertainty or incomplete information.[40] Therefore, unlike conventional software, expert systems are more tolerant of errors and imperfect knowledge.[41] They are "journey-oriented": Using heuristic knowledge-based rules and user input, the expert system operates in a nonlinear way to reach a solution.[42] The user, by choosing inputs that trigger various branches in the program, acts interactively with

---

29. Tuthill, *supra* note 19, at 48.

30. Chien & Liebowitz, *supra* note 6, at 2.

31. *Id.* at 3. Heuristics are strategies that are used to guide a search process to a solution. *Id.*

32. *Id.*

33. *Id.*

34. *Id.*

35. In this comment, the terms artificial intelligence and expert systems are used interchangeably because the legal issues associated with each of them apply to the other.

36. Chien & Liebowitz, *supra* note 6, at 10. Expert systems are also known as knowledge-based expert systems and expert problem-solvers. *Id.* Other subsets of AI include knowledge-based systems, intelligent tutoring systems, intelligent databases, and intelligent job aids. Tuthill, *supra* note 19, at 48.

37. Reece, *supra* note 8, at 24.

38. *Id.*

39. *Id.*

40. Chien & Liebowitz, *supra* note 6, at 15.

41. *Id.* at 10.

42. Tuthill, *supra* note 19, at 48.

the expert system.[43] Thus two different users, operating from the same fact base, might select different alternatives and generate different results.

Designing a functional expert system requires the interaction of a number of people: Domain (human) experts, knowledge engineers, programmers, program designers and developers.[44] A critical part of an expert system is the knowledge base,[45] that section of the program containing the facts and rules provided by a human expert.[46] An expert system "is only as good as its knowledge base."[47] The quality of the knowledge base depends on the expertise of both the domain expert and the knowledge engineer.[48] The domain expert provides the facts and determines the reasoning process, and the knowledge engineer, although not an expert in the particular subject area, builds an expert system by converting the knowledge into rules suitable for use in the system.[49] Once the rules are specified, programmers[50] write the actual code for the system.[51] Thus the development of the program is interactive, relying on a number of people of differing skills to produce something to be sold by a vendor.

Expert systems were traditionally custom-designed by universities and research companies for such activities as medical diagnosis, data and electrical analysis, automatic programming, and planning.[52]

---

43. *Id.*

44. *Id.* at 46-48; Reece, *supra* note 8, at 24.

45. An expert system consists of a dialog structure to provide a language interface for interaction of the user with the program, an inference engine allowing generation of a hypothesis to solve the problem, and a knowledge base consisting of facts and rules provided by a human expert. Chien & Liebowitz, *supra* note 6, at 11-12. Successful expert systems have been created when (1) there was at least one expert in the problem domain, (2) the expertise was based on judgment and experience, (3) the expertise was communicated to a knowledge engineer, (4) the problem was defined, (5) there was a consensus in the problem domain, and (6) test data were available. *Id.* at 10.

46. Reece, *supra* note 8, at 24.

47. Tuthill, *supra* note 19, at 48.

48. *Id.*; Reece, *supra* note 8, at 24.

49. Tuthill, *supra* note 19, at 48; Reece, *supra* note 8, at 24.

50. Reece, *supra* note 8, at 24. Designers specify the concept of the entire system; developers identify user needs, target markets, and specify the audience for the program. Tuthill, *supra* note 19, at 46.

51. "Code" is the generic term for both the languages and symbols used in computer programs. GEMIGNANI, *supra* note 20, app. at 658. Two types of instructions are generally thought to comprise code. Machine language, the instructions that can be directly executed by the central processing unit, is not readily understood by even skilled computer experts. *Id.* at 662. Source code is the high level language form of the program. *Id.* at 665. When assembled, a source code becomes an object code. *See id.* at 663. Source code is generally a human-readable form of code. Roland B. Desilets, Jr., Note, *Software Vendors' Exposure to Products Liability for Computer Viruses*, 9 COMPUTER/L.J. 509, 524 (1989).

52. Chien & Liebowitz, *supra* note 6, at 16-18.

As the systems have developed, however, they have fallen into three distinct categories: Those sold in a mass-marketing mode[53] as a ready-to-use program (i.e. a "turnkey" system); those custom-made for a particular application; and those modified for a particular user's needs.[54] The interaction between the expert system developers and the end-user is minimal for the first category, maximum for the second category, and at a moderate level for the third category.[55] For custom systems, the user may be the domain expert, and possibly even the knowledge engineer.[56] As such, the user is responsible for creating and loading the knowledge base.[57] This action blurs the distinction between the software developer and the end-user.

Mass-marketed expert systems can be categorized in a number of ways depending on their design and intended use. Some systems are sold as "products" for use any time by the buyer; others are provided as "services" for temporary use to a user who pays a fee.[58] In this latter case, the program stays under the management of the creator or vendor.[59] Additionally, the program may be either freestanding or embedded.[60] If freestanding, the expert system constitutes the nucleus of the application; little other software is needed for operation of the program.[61] If embedded, the expert system comprises only a part of the entire program and serves to act only when invoked for a specific purpose.[62]

3. *Sources of Program Error*

Computer systems are complex and provide fertile ground for error. In addition to problems arising from incorrect data entry, hardware failure or electrical noise,[63] errors or "bugs" can develop

---

53. Mass-market expert systems are defined as those applications that "generally run on microcomputers, cost less than $500, and have a limited range of expert performance." Eliot, *supra* note 5, at 9.

54. Reece, *supra* note 8, at 28.

55. The turnkey system has properties most like a product, the custom system most resembles a service, and the modified system is a hybrid of a product and a service. *Id.* at 28; *see infra* notes 110-16 and accompanying text.

56. Hayes, *supra* note 3, at 115.

57. *Id.*

58. Eliot, *supra* note 5, at 10. The distinction here between a "product" and a "service" may not be the same as that made when considering the applicability of U.C.C. Article 2. *See infra* notes 133-40 and accompanying text.

59. Eliot, *supra* note 5, at 10.

60. *Id.*

61. *Id.*

62. *Id.*

63. Michael C. Gemignani, *More on the Use of Computers by Professionals*, 13

from mistakes made by the programmer while preparing the code or flaws that occur during duplication of the disks that carry the code.[64] These types of errors are common to both conventional software and AI.[65] Expert systems, however, are subject to other errors, ranging from the unsuitability of the use of AI for a particular problem due to complexity or the need for common sense, to the lack of adequate expert knowledge or the failure of the human expert to supply accurate and complete facts and rules.[66] Additional problems can arise when the user implements the system: A poor choice of program, unrealistic reliance on the output, or faulty input.[67] Furthermore, if the system is not maintained and updated, the knowledge base may become obsolete, thus producing outdated output.[68]

With conventional software, the source of most errors can generally be determined because the error is reproducible.[69] Other errors resulting from transient conditions[70] may prove difficult to detect and understand. With expert systems, however, because of the interactive nature of the program with the user and the nonlinear approach to output,[71] it may not be possible to precisely determine how an error occurs.

4. *Summary*

Despite the fact that both conventional software and AI software can be encoded in computer language and run on conventional computing equipment, there are significant differences.[72] AI and expert systems manipulate knowledge; conventional software manipulates data.[73] The systems therefore vary in their development strategy, problem-solving methods, and level of user interaction.[74]

RUTGERS COMPUTER & TECH. L.J. 317, 323 (1987).

64. Reece, *supra* note 8, at 25.

65. *Id.*

66. *Id.* One author has divided expert system errors into two categories: Those resulting from poor design (e.g. unrealistic concept, insufficient or out-of-date knowledge base, incorrectly identified user group, and poor documentation) and those resulting from poor execution (e.g. bugs in the inference engine, incorrect links or branches in the program, inaccurate heuristic knowledge, and user input errors). Tuthill, *supra* note 19, at 46-47, 51.

67. Reece, *supra* note 8, at 25.

68. *Id.* at 25-26; Tuthill, *supra* note 19, at 47.

69. Gemignani, *supra* note 63, at 323 n.31.

70. Transient conditions include power surges and incipient defects in the equipment. *Id.*

71. Tuthill, *supra* note 19, at 48.

72. *Id.* at 48.

73. Reece, *supra* note 8, at 24.

74. Tuthill, *supra* note 19, at 48.

Unlike conventional software, in which output from given data is reproducible, with expert systems, the combination of a large number of program branches and a nonlinear approach means the user can influence the output.[75]

B. *Liability Standards*

One commentator stated, "[n]o complex computer program has ever been marketed that did not have some defect, somewhere."[76] Developers and vendors of artificial intelligence systems, which may be the most complex programs of all, must be concerned with potential liabilities.[77] Liability may arise under three scenarios: When remote parties such as a manufacturer and a consumer are connected by virtue of the sale of a product; when two parties are in a direct contractual relationship; and when the user relies on information supplied by the computer system.[78]

The first scenario considers negligence and strict liability under tort law. Breach of warranties, both express and implied, falls within both the first and the second scenarios. The third scenario again deals with negligence.

1. *Negligence*

Liability for negligence occurs in two situations with respect to computer programs: When the software is defective and when a party is injured as a result of using the software.[79] Both situations raise issues for AI programs. Negligence is the failure to use the care a reasonably prudent person would use under similar circumstances.[80] To prevail on a negligence claim a plaintiff must show the defendant had a duty of care, breached that duty, and caused an injury to the plaintiff as a result of that breach.[81] While there is little question a software vendor owes a duty of care to a consumer, the more difficult issue is what standard of care is owed.[82] It is well established that a seller of goods[83] has a duty to exercise the care of a

---

75. *Id.*

76. Hayes, *supra* note 3, at 115 (quoting Robert Cesari).

77. Tuthill, *supra* note 19, at 49-51.

78. NIMMER, *supra* note 24, ¶ 7.01, at 7-2.

79. *See* Tuthill, *supra* note 19, at 49.

80. BLACK'S LAW DICTIONARY 1032 (6th ed. 1990).

81. W. PAGE KEETON ET AL., PROSSER AND KEETON ON THE LAW OF TORTS § 30, at 164-65 (5th ed. 1984).

82. Gemignani, *supra* note 10, at 189.

83. See discussion of whether AI and expert systems constitute "goods" *infra* notes 110-

reasonable person to see that the goods sold do not harm the buyer.[84] Instead of this general standard, it has been proposed that the correct standard of care for the software vendor, and particularly for those creating the software, is that of a professional.[85] A professional is one who possesses a standard minimum of special knowledge and ability, and who undertakes work requiring special skill.[86] A professional is held to a higher standard than a non-professional, and the performance of a professional is measured against an ordinary member of the profession who has the same level of knowledge, training, and skill.[87] Thus computer professionals (assuming the "professional" designation is appropriate) would be comparable to other professionals such as doctors, engineers, and lawyers whose professional breach is called malpractice.[88] Courts have been unwilling to apply such a standard,[89] perhaps because of the lack of a licensing procedure for programmers (that would establish a gauge of minimum ability),[90] or because of the fact that programming is only a small part of the total process of software development.[91] The problem of establishing a standard of care for AI is complicated by the

---

16 and accompanying text.

84. KEETON ET AL., *supra* note 81, § 96, at 684. Manifestations of a seller's lack of care include misrepresentation of the type of goods or their fitness for a particular purpose, failure to disclose potential danger if the good is used for the buyer's purpose, sale to an obviously incompetent buyer, and failure to exercise reasonable care to inspect the goods. *Id.*

85. Susan Nycum, *Liability for Malfunction of a Computer Program*, 7 RUTGERS J. COMPUTERS, TECH. & L. 1, 9 (1979).

86. KEETON ET AL., *supra* note 81, § 32, at 185.

87. Tuthill, *supra* note 19, at 50.

88. KEETON ET AL., *supra* note 81, § 32, at 185-88.

89. Chatlos Systems Inc. v. National Cash Register Corp., 479 F. Supp. 738 (D.N.J. 1979) (declining to create a cause of action for computer malpractice under state law), *rev'd on other grounds*, 635 F.2d 1081 (3d Cir. 1980), *aff'd after remand*, 670 F.2d 1304 (3d Cir. 1981), *cert. dismissed*, 457 U.S. 1112 (1982); Triangle Underwriters, Inc. v. Honeywell, Inc., 604 F.2d 737, 741, 744-45 (2d Cir. 1979) (rejecting cause of action for computer malpractice because a professional relationship did not exist when plaintiff attempted to avoid a statute of limitations problem by arguing that the "continuous treatment" doctrine used in medical malpractice should extend to negligent repairs); Invacare Corp. v. Sperry Corp., 612 F. Supp. 448 (N.D. Ohio 1984). In *Invacare*, after holding that the computer malpractice claim was nonactionable, the court held that computer professionals were subject to the same negligence standard as machinists, electricians, carpenters, blacksmiths, and plumbers. *Id.* at 453. The court did not consider whether computer malpractice involved a higher standard of care. *Id.* at 454. *But see* Data Processing Servs., Inc. v. L.H. Smith Oil Corp., 492 N.E.2d 314 (Ind. Ct. App. 1986) (holding that a contract to provide computer programming was a service contract and that programmer breached implied promise of having reasonable skill and diligence possessed by well-informed members of the trade). *See generally* Sue G. Graziano, *Computer Malpractice—A New Tort on the Horizon?*, 17 RUTGERS COMPUTER & TECH. L.J. 177 (1991).

90. Nycum, *supra* note 85, at 9-10.

91. Gemignani, *supra* note 10, at 190.

fact that different standards may be appropriate for the different people involved in developing the system[92] and it might be very difficult to detect which of them created the fault. Certainly for the human expert, specifically selected due to the high level of skill and expertise possessed, a high level of care is expected, especially given the potential for catastrophe with a defective program.[93] The hybrid role of the knowledge engineer, who acts as a bridge between the human expert and the programmer, suggests that a professional standard should be applied because it is the expertise that is critical to the success of the program.[94] Other members of the development team, such as a quality assurance technician, might be held to a lower standard.

A breach of the duty of care may occur in a number of ways. If the software is defective due to errors and malfunctions that could have been detected by the vendor and/or creator, the key issue is how much testing is enough.[95] For AI systems this is an especially difficult problem because not all permutations and branches of a program can ever be tested, particularly if the user has the freedom to make choices during the normal operation of the program. However, as discussed above,[96] if an expert system is inadequate for a particular problem because common sense is needed for a solution, if the knowledge base is incomplete or inadequate, or if the problem is too complex, the development, marketing, and use of the system could be negligent.[97] Errors may be generated by the vendor due to incorrect information supplied by the human expert, poor design, manufacturing, testing, and distribution, inaccurate or inadequate warnings and documentation, or failure to maintain and update the knowledge base.[98] The user also may act negligently by providing faulty input or selecting the incorrect program for the task.[99] Furthermore, if the user unduly relies on the output and does not exercise sufficient

---

92. Reece, *supra* note 8, at 24.

93. *Id.* at 25.

> [O]ne of the most alarming features of expert systems is that they can dramatically multiply the potential for harm that can be caused by a single human expert because any errors or omissions by the expert will become an integral part of the system to be applied by hundreds or even thousands of users.

Id.

94. *See id.* at 25.

95. Gemignani, *supra* note 10, at 191.

96. *See generally supra* notes 63-68 and accompanying text.

97. *See* Reece, *supra* note 8, at 25; Tuthill, *supra* note 19, at 51.

98. Reece, *supra* note 8, at 25, 28.

99. *Id.* at 25.

judgment in evaluating the answers, the user may be negligent.[100]

The causation element also must be considered. To prevail on a negligence claim, the plaintiff must show there is "some reasonable connection between the act or omission of the defendant and the damage which the plaintiff has suffered."[101] Due to the number of potentially negligent parties and the complexity of the program, proving that the program actually caused the injury may be especially difficult.[102] Furthermore, the vendor or creator of the software may assert a defense to negligence.[103] The user may be held to have assumed the risk,[104] or at least contributed to the negligence[105] in one of two ways: First, by failing to maintain a level of skill necessary for reasonably prudent use of the program;[106] and second, by misplacing confidence in the infallibility of the program with the consequence that the results are not carefully reviewed and tested by the user.[107]

2. *Strict Liability*

Strict liability[108] does not require the plaintiff prove the defend-

---

100. Tuthill, *supra* note 19, at 47, 50.

101. KEETON ET AL., *supra* note 81, § 41, at 263. Causation includes cause in fact and proximate cause (legal cause). *Id.* "[C]ausation in fact extends not only to positive acts and active physical forces, but also to pre-existing passive conditions which have played a material part in bringing about the event." *Id.* § 41, at 265.

102. *See id.* § 41, at 265.

103. Defenses to negligence include contributory negligence, comparative negligence, and assumption of risk. *Id.* §§ 65-68, at 451-80.

104. Assumption of risk includes situations where the plaintiff consents in advance to relieving the defendant of an obligation and agrees to risk injury resulting from the defendant's act or omission; where the plaintiff voluntarily tacitly consents to the negligence with the knowledge defendant will provide protection; and where the plaintiff voluntarily proceeds to act after being informed of the risk. *Id.* § 68, at 480-90.

105. Contributory negligence is conduct by the plaintiff that results in harm to the plaintiff and is below the standard required for the plaintiff's own protection. *Id.* § 65, at 451.

106. Tuthill, *supra* note 19, at 50.

107. Roy N. Freed, *A Lawyer's Guide Through the Computer Maze*, 6 PRAC. LAW. 15, 40 (Nov. 1960), *reprinted in* COMPUTERS AND LAW: A REFERENCE WORK 2, 14 (Roy N. Freed ed., 5th ed. 1976).

108. Strict liability is a theory applied "in product liability cases in which seller is liable for any and all defective or hazardous products which unduly threaten a consumer's personal safety." BLACK'S LAW DICTIONARY 1422 (6th ed. 1990). No privity of contract is required between the seller and a third party who is injured. KEETON ET AL., *supra* note 81, § 100, at 703-04. The theory is stated in RESTATEMENT (SECOND) OF TORTS § 402(A) (1964):

> (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

ant was negligent or at fault, but rather the product was defective and unreasonably dangerous when used in a normal, intended, or reasonably foreseeable manner, and that the defect caused plaintiff's injury.[109] Because strict liability applies to "any product,"[110] the first issue in applying strict liability to software is determining whether the software is classified as a product (in which case strict liability may apply) or a service (in which case strict liability does not apply).[111] While the extremes are easily classified,[112] many expert systems are so complex in design and function as to be hybrids.[113] It has been proposed that the program be classified by analyzing its function and its end product.[114] Thus, if the expert system is used to provide a service that a human might perform, e.g. investment counseling, the program might be designated a service.[115] Conversely, if the system merely provides routine data analysis, the program might be designated a product. This "function" approach suffers from the fact that many expert systems designed to provide such "services" are themselves mass-marketed in the way that "goods" routinely are sold.[116] In addition, analysis of function and end product includes subjective evaluation, producing inconsistent results in classification.

The second issue in determining whether strict liability applies is determining whether the software was defective and thus unreasonably unsafe.[117] A product is considered defective when it is correctly made according to an unreasonably dangerous design (i.e. a design defect),[118] when there is incorrect implementation of a safe

---

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.

Id.

109. RESTATEMENT (SECOND) OF TORTS § 402(A) (1964).

110. *Id.*

111. Tod M. Turley, Note, *Expert Software Systems: The Legal Implications*, 8 COMPUTER/L.J. 455, 457 (1988).

112. Mass-produced, mass-marketed programs, both conventional and expert systems, are likely to be classified as a good or a product and thus be subject to strict liability; custom-generated software with unique features is likely to be classified as a service and thus be subject to a negligence standard. *Id.*

113. Reece, *supra* note 8, at 28.

114. Turley, *supra* note 111, at 457-58.

115. *Id.* at 458 n.16.

116. *See* Eliot, *supra* note 5, at 9.

117. RESTATEMENT (SECOND) OF TORTS § 402(A) (1964).

118. KEETON ET AL., *supra* note 81, § 99, at 695.

design (i.e. a manufacturing defect),[119] or when there are inadequate warnings and instructions concerning the potential dangers of use.[120] Of particular concern are unavoidably dangerous products that cannot be made safe given the present state of human skill and knowledge.[121] For complex expert system software that probably cannot be fully tested,[122] this latter concern is important.

Two additional factors associated with strict liability must be considered. First, in order for conventional strict liability to apply, a physical harm, i.e. personal injury or property damage, must result from use of the product.[123] Therefore, strictly economic loss is generally insufficient to establish liability.[124] If defective software causes customer loss or poor business decisions, strict liability does not apply.[125] Second, unlike contract warranties,[126] strict liability cannot be disclaimed.[127] In addition, warnings may not be deemed sufficient as protection against strict liability, especially if they are buried in a lengthy users' manual.[128]

Applying strict liability to expert system software is particularly relevant for two reasons.[129] First, the definition of product has been broadened beyond tangible property,[130] thus increasing the possibility it applies to software. Secondly, applying strict liability serves public policy considerations such as risk-spreading.[131] Strict liability,

---

119. Birnbaum, *supra* note 9, at 138-39.

120. KEETON ET AL., *supra* note 81, § 99, at 695.

121. *Id.* at 700-01.

122. *See* Tuthill, *supra* note 19, at 48.

123. Birnbaum, *supra* note 9, at 140; *see also* KEETON ET AL., *supra* note 81, § 101, at 708.

124. *See* KEETON ET AL., *supra* note 81, § 101, at 708.

125. Birnbaum, *supra* note 9, at 140.

126. *See infra* notes 133-53 and accompanying text.

127. GEMIGNANI, *supra* note 20, § 29:2, at 413.

128. Birnbaum, *supra* note 9, at 139.

129. Reece, *supra* note 8, at 24.

130. *See* Ransome v. Wisconsin Elec. Power Co., 275 N.W.2d 641, 647-48 (Wis. 1979) (holding that electricity is a consumable product and thus strict liability applies). A conventional definition of product is "[g]oods produced or manufactured, either by natural means, by hand, or with tools, machinery, chemicals, or the like." BLACK'S LAW DICTIONARY 1209 (6th ed. 1990).

131. Gemignani, *supra* note 10, at 196-97. Among the reasons suggested for application of strict liability, four are generally recognized. First, responsibility for damages due to defective goods should be borne by the party in the best position to detect and eliminate the defects. Second, the party best able to absorb and spread the risk through insurance should bear the liability. Third, the injured party should not have to meet burdensome proof requirements in order to obtain relief. Fourth, the doctrine of *caveat emptor* is of little importantance in view of modern marketing methods. David A. Hall, Note, *Strict Products Liability and Computer Software: Caveat Vendor*, 4 COMPUTER/L.J. 373, 373 n.1 (1983); *see also* Susan Lanoue, Comment, *Computer Software and Strict Products Liability*, 20 SAN DIEGO L. REV. 439,

however, has not been successfully applied by the courts for software applications or for situations in which the project was considered a professional service.[132]

3. *Breach of Warranties*

A third area of liability to consider is that of warranties: what warranties, if any, are expressed or implied in the sale of software?[133] If software is defined as a good, then both the express and implied warranties described in Article 2 of the Uniform Commercial Code (hereinafter U.C.C.) apply.[134] Thus, prior to applying the standards of the U.C.C., the distinction between products (goods) and services, as discussed above, must be made.[135] Because a computer program can be moved and conveyed to another at the time of sale, it is arguably a good.[136] However, if the program, like many expert systems, is a hybrid, such a designation may not be strictly accurate.[137] In at least one case,[138] a court used the "predominant feature" test to balance the relative service and product aspects of a contract for the purchase of software.[139] In that case the court found the vendor's contractual obligations to install the software, debug the system, and provide training were services subservient to the sale of the product, and contractual remedies were applied.[140]

447-49 & n.12 (1983). The Lanoue comment describes the goals of strict liability as loss spreading, accident reduction, victim compensation, and loss compensation. *Id.* These policies were first discussed in Justice Traynor's concurring opinion in Escola v. Coca Cola Bottling Co., 150 P.2d 436, 440-43 (Cal. 1944).

132. *See* Chatlos Systems Inc. v. National Cash Register Corp., 479 F. Supp. 738, 740-41 & n.1 (D.N.J. 1979), *rev'd on other grounds*, 635 F.2d 1081 (3d Cir. 1980), *aff'd after remand*, 670 F.2d 1304 (3d Cir. 1981), *cert. dismissed*, 457 U.S. 1112 (1982); *see also* La Rossa v. Scientific Design Co., 402 F.2d 937 (3d Cir. 1968) (finding that strict liability did not apply to an engineering company that designed a plant containing a reactor incorporating carcinogenic vanadium pellets because engineering was a professional service).

133. Desilets, *supra* note 51, at 513. This comment does not discuss the issue of whether a software program is "licensed" rather than "sold."

134. U.C.C. § 2-102 (1990). Article 2 applies to "transactions in goods." *Id.* Goods are "all things (including specially manufactured goods) that are movable at the time of identification to the contract for sale." *Id.* § 2-105(1).

135. Desilets, *supra* note 51, at 513.

136. Lanoue, *supra* note 131, at 443 n.12.

137. The service aspects, such as delivery, installation, and program start-up, which are incidental to the sale of the software, are not considered sufficient to automatically characterize the software as a service and thus remove it from the scope of the U.C.C. SOMA, *supra* note 16, § 3.07, at 78-79 & n.21.

138. RXX Indus., Inc. v. Lab-Con, Inc., 772 F.2d 543 (9th Cir. 1985).

139. *Id.* at 546.

140. *Id.* The emphasis in this case was on breach of contract issues, and the issue of strict liability was not raised. *See also* Turley, *supra* note 111, at 459.

Express warranties fall under the provisions of U.C.C. section 2-313, which provide that any fact or promise by the vendor to the customer relating to the goods sold creates an express warranty that the goods shall meet the promised standard.[141] Express warranties include specific written promises made as part of the contract, as well as oral representations made by a salesman, promotional brochures, instruction booklets, and advertisements.[142] Because no software vendor would presume to sell a defect-free program,[143] it is common for the warranty to be qualified by disclaimers, limitations, and modifications.[144] Such disclaimers are generally upheld if they are not unconscionable under the particular circumstances,[145] and if the language of the disclaimer is consistent with the warranty.[146]

Given that few unqualified express warranties are made by vendors of software, the implied warranties of merchantability and fitness of use for a particular purpose should be considered.[147] The implied warranty of merchantability presented in U.C.C. section 2-314 allows a purchaser to have confidence the vendor is selling goods that are of ordinary quality in the trade and that conform to any promises made on the label.[148] The implied warranty of fitness for a particular use under U.C.C. section 2-315 imposes an additional duty on the vendor if it is known at the time of sale or contract what specific purpose the buyer intends for the product, and if the buyer has relied on the vendor's skill or judgment to sell the appropriate product.[149] This warranty is critical for expert systems if the program is sold with the idea—expressed or implied—that it will provide a "total solution" to the customer.[150] Furthermore, the user may not have adequate knowledge of computers and expert systems and will rely heavily on the vendor's "expertise."[151] These implied warranties can be restricted by the use of disclaimers under the provi-

---

141. U.C.C. § 2-313 (1990).

142. Gemignani, *supra* note 10, at 179 n.22.

143. Desilets, *supra* note 51, at 515.

144. NIMMER, *supra* note 24, ¶ 6.07[1], at 6-20. Nimmer notes that because the express warranty is part of the basis of the bargain, in theory, it cannot be disclaimed. Pro forma language disclaiming additional representations is insufficient. *Id.*

145. Tuthill, *supra* note 19, at 49.

146. NIMMER, *supra* note 24, ¶ 6.07[1], at 6-20.

147. Desilets, *supra* note 51, at 515.

148. U.C.C. § 2-314 (1990).

149. *Id.* § 2-315.

150. Desilets, *supra* note 51, at 516. One writer pointed out "expert system technology usually enhances a product's suability and differentiates it in the public's mind." Eliot, *supra* note 5, at 9-10.

151. Desilets, *supra* note 51, at 515-16, 524.

sions of U.C.C. section 2-316.[152] If the disclaimers are part of a fairly negotiated contract, meet U.C.C. guidelines, and are not against public policy, they are generally upheld.[153] These issues are important for expert systems that have been mass-marketed and for which there is no individually negotiated sales contract.

III. ANALYSIS

A. *Liability Options*

The limited case law concerning software[154] has focused on conventional programs.[155] Although a standard for liability is not established, these cases are useful guides, and, when appropriate, can be applied to AI software. However, the differences between expert systems and conventional software, as well as the differences between mass-marketed and custom-designed software, mean a standard for conventional software cannot be applied unilaterally. Thus, the effects of applying various liability standards to AI and expert system software as compared to conventional software are considered.

1. *Apply a Strict Liability Standard to All Software*

From the perspective of the software consumer, strict liability is the most attractive standard of liability for recovery in the event of a defect. This theory does not demand proof that the software developer, programmer, or vendor (collectively "the software dealers") was at fault.[156] If such a requirement were imposed, it would be particularly difficult for the average computer user, who may be a "technical[] illiterate,"[157] unable to search machine code to locate a defect.[158] Without question, the software dealers are better able to detect a fault: they are in possession of the source code,[159] and they employ skilled workers who have experience in the field.[160] With their technical sophistication, software dealers are better positioned

---

152. U.C.C. § 2-316 (1990).

153. Desilets, *supra* note 51, at 516-17.

154. *See supra* notes 89, 132, 138 and accompanying text.

155. *See generally* Birnbaum, *supra* note 9; Gemignani, *supra* note 10; Nycum, *supra* note 85; Desilets, *supra* note 51; Hall, *supra* note 131; Lanoue, *supra* note 131; Turley, *supra* note 85.

156. *See* RESTATEMENT (SECOND) OF TORTS § 402(A) (1964).

157. Birnbaum, *supra* note 9, at 145.

158. Desilets, *supra* note 51, at 524.

159. *Id.*

160. Nycum, *supra* note 85, at 17.

to determine whether there are risks in using the software, whether those risks can be prevented, and what procedures are necessary to eliminate the problems.[161]

Strict liability is only appropriate if the software is considered a product, thus falling under the "any product" language of section 402(A)(1) of the *Restatement (Second) of Torts*.[162] Arguably, the tangible disk containing the software is a product, something, analogous to the U.C.C.'s definition of "goods," which is "movable at the time of . . . sale."[163] Furthermore, software can be considered "as the completion of an incomplete machine,"[164] the component that must be added to, and work in conjunction with, hardware to provide a functional computer.[165] If the hardware meets the requirements for a product, so should the "component" software. Indeed, one commentator contends it makes no sense to have a system where the form in which the software is provided dictates the standard of liability when that software causes an injury.[166] Should there be any difference in liability between software supplied as a computer-installed module (i.e. part of the computer hardware and thus a "product") or software supplied on a floppy disk (i.e. potentially not a "product")?[167] Arguably not, because both forms of delivery convey the same information.

Even if software is not considered a tangible product, the application of strict liability still should not be precluded. In *Ransome v. Wisconsin Electric Power Co.*,[168] the court held that strict liability could apply to damage caused by an intangible entity, i.e. electricity. Electricity was found to be a consumable product,[169] "a form of energy that can be made or produced by men, confined, controlled, transmitted and distributed."[170] Despite the fact that the transmission of electricity is a service, the electricity itself was found to be a product.[171] Certainly software is human-produced, controlled, and distributed. Although it is not consumed in the conventional sense,[172]

---

161. GEMIGNANI, *supra* note 20, § 29:17, at 421-22.
162. RESTATEMENT (SECOND) OF TORTS § 402(A) (1964).
163. U.C.C. § 2-105(1) (1990).
164. GEMIGNANI, *supra* note 20, § 29:17, at 422.
165. *Id.*
166. *Id.*
167. *Id.*
168. Ransome v. Wisconsin Elec. Power Co., 275 N.W.2d 641 (Wis. 1979).
169. *Id.* at 648.
170. *Id.* at 643.
171. *Id. See generally* Hall, *supra* note 131, at 389.
172. "To expend (fuel, for example); use up." THE AMERICAN HERITAGE DICTIONARY OF THE ENGLISH LANGUAGE 286 (William Morris ed., 1980).

the fact that software can be used repeatedly and has the ability to benefit society[173] further supports the idea that it is a product.

It can be argued by analogy with other products that software should be subject to strict liability because an error in the program constitutes a design defect.[174] In the same way an error in a blueprint for a car constitutes a design defect in the finished vehicle, a bug in a program results in an error when the program is run.[175] Because the auto manufacturer can be held strictly liable for the flaw, the software vendor should likewise be held liable for a program error.[176]

Policy reasons also dictate that strict liability should be applied to software. A key consideration in the application of strict liability is the relative position of the victim with respect to the defendant.[177] Applying strict liability allows the financial burden to be placed on the manufacturer and/or the vendor, the parties most able to bear the costs of the loss.[178] The manufacturer is also in a better position to detect and correct flaws in the program, thus contributing to accident reduction.[179] Fairness requires that compensation be provided to the innocent victim who has been financially damaged because of the injury.[180] This compensation can be supplied by the manufacturer, who is in the better financial position relative to the victim. Furthermore, the manufacturer can absorb the costs, either through insurance or price adjustments.[181]

Policy reasons also dictate that retailers of the software be held to a strict liability standard. Like manufacturers, retailers are in a more favorable position to bear the costs of an accident than the innocent purchaser.[182] Imposing such a high burden furthers the policy goal of accident prevention in three ways: First, by encouraging retailers to deal with manufacturers who design and construct safe products; second, by providing financial security for those injured by a defective product; and third, by relieving the victim of the heavy

---

173. Birnbaum, *supra* note 9, at 145.

174. Nycum, *supra* note 85, at 17. *But see* Roy N. Freed, *Products Liability in the Computer Age*, 17 JURIMETRICS J. 270, 275-79 (1977). Freed contends that a program is a process, not a product, and to apply strict liability is thus improper. *Id.*

175. Nycum, *supra* note 85, at 17-18.

176. *Id.*

177. Lanoue, *supra* note 131, at 448.

178. *Id.* at 448 n.39. This policy is known as "loss spreading" and is based on the premise that the manufacturer will distribute the costs through higher prices to customers. *Id.*

179. *Id.*

180. *Id.*

181. *Id.*

182. KEETON ET AL., *supra* note 81, § 100, at 706.

burden of proving where in the manufacturing chain the defect originated.[183] In addition, holding the retailer to a strict liability standard provides recourse to the injured consumer in the event the manufacturer is insolvent.[184] Finally, due to the nature of the continuing relationship between the retailer and the manufacturer, the retailer is more likely to successfully resolve the incident with the manufacturer without litigation than is the victim.[185]

Two other policy reasons have been proposed to support the application of strict liability to software.[186] First, strict liability provides a means of recovery for a plaintiff injured as the result of "spontaneous malfunctions" that can occur without evidence of any negligence.[187] If such a malfunction occurs, no recovery is possible under a negligence theory.[188] Second, it avoids random application of the law.[189] If an injured driver could collect under strict liability for a defective steering mechanism, it is unreasonable to forbid recovery under the same theory for the same injuries which were caused as a result of a defectively programmed on-board computer.[190]

Despite the benefits of applying strict liability to software, arguments have been proposed against such application.[191] These include the position that software is not a product,[192] and therefore does not fall under section 402(A) of the *Restatement (Second) of Torts*.[193] It has been suggested that section 402(A) does not apply if the product is expected "to be processed or substantially changed before it reaches the user."[194] Software, both conventional and expert systems, does "change" when it is loaded into a system from a disk or when it is translated from source code to machine language.[195] However, this change is certainly expected by the vendor: the software would not work without it.[196] Change also occurs with expert systems. These

---

183. *Id.* at 707.

184. *Id.* at 706.

185. *Id.*

186. Lanoue, *supra* note 131, at 449.

187. *Id.* at 449 n.46. How these defects occur is not fully understood. *Id.*

188. *Id.* at 449.

189. *Id.*

190. *Id.*

191. *See generally* GEMIGNANI, *supra* note 20, §§ 29:1-29:24, at 411-26; Turley, *supra* note 111, at 470-75.

192. Nycum, *supra* note 85, at 16.

193. RESTATEMENT (SECOND) OF TORTS § 402(A) (1964).

194. GEMIGNANI, *supra* note 20, § 29:3, at 414 (quoting RESTATEMENT (SECOND) OF TORTS § 402(A) (1964)).

195. *See id.*

196. *See id.*

systems are designed to work with user intervention as part of the expert system's deductive process.[197] The change, therefore, does not occur before the product reaches the user, but rather in conjunction with the user. While vendors may not be aware of every possible change that may be made,[198] they knowingly sell the expert system to accommodate such change.[199]

It has also been proposed that section 402(A) does not apply to software because it is not unreasonably dangerous.[200] Thus, while a rocket[201] or a nuclear power plant[202] might be unreasonably dangerous, the computer program that controls them is not. However, given that a defect in an expert system that is used in a medical application might lead to a death,[203] it appears such programs could be unreasonably dangerous.

2. *Apply a Negligence Standard to All Software*

The advantage of applying a negligence standard to software, both conventional and expert systems, is that "[t]here is no dispute that a vendor of products or services owes a duty of care to the consumer."[204] There are, however, two major problems with applying a negligence standard: First, "[t]he duty of care the software development industry owes to software users is not clear,"[205] and second, tracing the source of a program defect to prove it caused the plaintiff's injury is very difficult.[206]

To solve the first problem and define the duty of care for software developers, several authorities have argued that a professional standard should be imposed, thus paving the way for the tort of computer malpractice.[207] A number of difficulties exist with this approach.[208] First, although the nature of writing software programs is professional, there are neither established professional standards

197. Turley, *supra* note 111, at 463.
198. Tuthill, *supra* note 19, at 48.
199. *See generally id.* at 48-50.
200. GEMIGNANI, *supra* note 20, § 29:7, at 416.
201. *Id.*
202. Desilets, *supra* note 51, at 525.
203. Davis, *supra* note 9, at A1.
204. Gemignani, *supra* note 10, at 189.
205. Desilets, *supra* note 51, at 518-19.
206. *Id.* at 520.
207. Graziano, *supra* note 89, at 182-86; Kevin S. MacKinnon, *Computer Malpractice: Are Computer Manufacturers, Service Bureaus, and Programmers Really the Professionals They Claim to Be?*, 23 SANTA CLARA L. REV. 1065, 1066-74 (1983); Nycum, *supra* note 85, at 8-10.
208. Desilets, *supra* note 51, at 519.

for practice nor education or licensing requirements.[209] Second, many mass-produced computer programs are arguably goods, not services.[210] Third, the tort of computer malpractice has found only limited acceptance by the courts,[211] and a number of courts have specifically dismissed it.[212] There is no evidence there is a substantial difference between conventional and expert system software for any of these problems, although the contributions of many different people during the development of expert systems suggests an additional problem. Different contributors, who have different educational and professional backgrounds,[213] might be subject to different standards.

The second problem with applying a negligence standard to software is that the plaintiff has the burden of proof to show that the defendant breached the duty of care and that the breach caused plaintiff's injury.[214] Satisfying this burden is particularly difficult for expert systems because they are "the embodiment of complex programming techniques which require input of expert information into the knowledge base and the structure of the knowledge base itself."[215]

3. *Consider the Method of Marketing*

a. *Mass-Marketed Programs*

Mass-marketed programs, either conventional or expert systems, fall readily into the strict liability category. Mass-marketed programs are mass-produced and promoted using conventional product marketing techniques: Television, trade journal, and newspaper advertising.[216] Programs are delivered as a finished product and purchased ready to use, off the shelf, in computer stores, or by mail, frequently by computer or fax order without any apparent human intervention.[217] Prepackaged and sealed disks preclude customer

209. *Id.*

210. *Id.*

211. Data Processing Servs., Inc. v. L.H. Smith Oil Corp., 492 N.E.2d 314 (Ind. Ct. App. 1986).

212. Chatlos Sys. Inc. v. National Cash Register Corp., 479 F. Supp. 738 (D.N.J. 1979), *rev'd on other grounds*, 635 F.2d 1081 (3d Cir. 1980), *aff'd after remand*, 670 F.2d 1304 (3d Cir. 1981), *cert. dismissed*, 457 U.S. 1112 (1982); Triangle Underwriters, Inc. v. Honeywell, Inc., 604 F.2d 737, 745 (2d Cir. 1979); Invacare Corp. v. Sperry Corp., 612 F. Supp. 448 (N.D. Ohio 1984).

213. *See supra* notes 92-94 and accompanying text.

214. Desilets, *supra* note 51, at 518.

215. Turley, *supra* note 111, at 461.

216. Hall, *supra* note 131, at 392.

217. *Id.* at 393.

"test-drives," requiring the consumer to rely on the manufacturer's representations as to specifications and performance.[218] There is no opportunity to bargain with the software dealer concerning the program or its capabilities.[219] In the same way a car (the quintessential "product") is purchased without the buyer understanding precisely how it works, mass-marketed software is purchased without the buyer comprehending, controlling, or being concerned about its development.[220]

Although expert systems are fundamentally different from conventional software because they are capable of "interacting" with the user,[221] the arguments for using a strict liability standard for all mass-marketed software, both conventional and expert system software, prevail. One commentator does not agree, however, and proposed applying a negligence standard to expert systems.[222] The five elements listed in that proposal would make the product/service and the strict liability/negligence distinctions more difficult:

> (1) one-of-a-kind applications; (2) human intervention as part of the expert system's deductive process; (3) an application area where no reasonable user would *blindly* rely on the output of the expert system; (4) experimental programs where the user is aware of the infancy of the testing process; and (5) where the user has contracted [with] a programmer to develop an expert system and compensation is for the programmer's services rather than the value of the expert system program.[223]

None of these points, however, with the possible exception of the second, applies to mass-marketed expert systems. By definition, mass-marketed systems are not one-of-a kind applications, but are designed for wide use.[224] The current mass-marketed systems generally "have a limited range of expert performance"[225] and are directed to markets where the risks of misuse are presumed to be limited.[226] Thus, for these systems, even "blind" reliance is unlikely to create a serious problem. For those mass-marketed systems that have

---

218. *Id.* at 393-94.
219. SOMA, *supra* note 12, § 3.12, at 89.
220. Hall, *supra* note 131, at 393-94.
221. Tuthill, *supra* note 19, at 48; *see supra* notes 31-39 and accompanying text.
222. Turley, *supra* note 111, at 463.
223. *Id.*
224. SOMA, *supra* note 16, § 3.12, at 89.
225. Eliot, *supra* note 5, at 9.
226. Among those mass-marketed expert systems available as of December 1989 are TeckChek, which analyzes software programming abilities, RootDirectory, which provides gardening advice, and Personal Pro, which dissects a golfer's swing. *Id.* at 10, 13, 14.

great potential for substantial harm, e.g. expert systems that provide medical advice,[227] the emphasis in the analysis of strict liability should be on the "unreasonably dangerous" aspects of the program, rather than on the "product versus service" distinction. There is no evidence that experimental programs are mass-marketed, probably due to the need to monitor the results. Additionally, because of the methods employed for development and sale, there is never a specific contract between user and programmer with mass-marketed expert systems software.[228] Finally, in considering the second element of the proposal, the limited capacity of expert systems makes its likely user interaction also limited. Therefore, the human intervention factor of the second element does not detract from the application of strict liability for mass-marketed systems.

b. *Custom-Designed Programs*

Custom-designed or custom-modified software, either conventional or expert systems, does not easily fall into the strict liability category because, unlike the mass-marketed software, it does not have the attributes of a product. In general, a contract with a software designer or vendor to produce a special program is a service, analogous to a contract with an engineer or architect to design a bridge or a building. In both cases, the contractor is relying on the professional skill of the creator. As stated in *La Rossa v. Scientific Design Co.*, "Professional services do not ordinarily lend themselves to the doctrine of tort liability without fault because they lack the elements which gave rise to the doctrine. There is no mass production of goods . . . ."[229] This language emphasizes the importance of mass production in applying strict liability.

While it is possible to argue that even one-of-a-kind items such as custom-designed software should be held to a strict liability standard,[230] there is little precedent for this application for either conventional or expert system software. Although it may be easy to argue that a particular program, e.g. one controlling a nuclear power plant or an air traffic control system, is inherently "unreasonably dangerous" when used as intended and thus should be subject to strict liability, a better standard is that each program must be consid-

227. FamilyCare Software provides pediatric advice. *Id.* at 9.

228. *Id.* at 9-10.

229. La Rossa v. Scientific Design Co., 402 F.2d 937, 942 (3d Cir. 1968).

230. Lanoue, *supra* note 131, at 443-44. In that article the author notes that many products, such as cars, can be hand-crafted to meet the needs of the individual consumer, without converting the "product" into a "service". *Id.* at 444.

ered on a case by case basis.[231] Such an analysis requires that the entire scope of the project be reviewed—the reason for the design, the intent and contribution of the customer, the skill and reliability of the programmers, the testing of the program, and the contract provisions—before determining the liability of the various parties. Both conventional tort and contract remedies should be considered. In particular, contract remedies may be available to the user of a custom-designed program that are unavailable to a user of mass-marketed software. Breach of warranty claims are more likely to apply to custom-designed software because there is a greater chance that a specific contract exists between the vendor and the user. In addition, courts are more likely to regard contracts (and any associated disclaimers) as binding if they have been negotiated between parties of relatively equal bargaining power.[232]

This analysis applies equally well to a custom-designed expert system as it does to conventional software. However, the importance of considering all the factors relevant to liability is amplified in such an analysis. In a custom-designed system, it is relatively easy to identify the domain expert, the knowledge engineer, and the programmer, and assess their potential liabilities. Furthermore, it is possible to determine how much the user contributed to the design and development of the system. If the user acted as the domain expert and provided an inaccurate knowledge base, contracted for development of a system based on a faulty concept that was user-suggested, or placed undue reliance on the system,[233] contributory negligence must be considered.[234] Strict liability clearly should not be applied automatically in this kind of situation.

B. *Alternative Standards*

Because strict liability has not yet been applied to computer software, additional safeguards may be warranted to protect the pub-

---

231. In fact, for programs of this type and others, it has been suggested that using a potentially flawed program may be preferable to the alternatives: Either not proceeding with the project at all or relying on humans to monitor and make complex computations that are truly only feasible with a computer. Gemignani, *supra* note 63, at 322.

232. Turley, *supra* note 111, at 473.

233. Tuthill, *supra* note 19, at 46-47.

234. *See also* Nycum, *supra* note 85, at 13-14. In this article, it is proposed that contributory negligence should be considered when a customer fails to convey his needs accurately and completely to a programmer, when a customer fails to notify a programmer of a material change in conditions that would warrant revision of the program, when erroneous or obsolete data is used, when too much or too little reliance is placed on the computer, when there is no back-up system, or when the user ignored an obvious error. *Id.*

lic from the harm generated by software-derived computer failures. Three alternatives, all relevant for expert systems, should be considered: Regulation of software, regulation of software creators, and regulation of software users.

1. *Regulation of Software*

In attempting to screen out unsuitable and unreliable software, it has been proposed that programs designed for use in professions such as medicine, law, and accounting should be certified by the appropriate professional organization.[235] Thus an expert system used by medical professionals would be approved by the American Medical Association. If the program were "responsible" for recommending drug doses, licensing from the Food and Drug Administration might also be required.[236] Similar licensing by the Federal Aviation Administration would be appropriate for air traffic control software.[237] This certification would provide a national standard for expert systems that would furnish users with confidence that a minimum level of reliability was present and provide developers and vendors with guidelines for product development. In addition, certification could be a positive sales tool for software vendors by providing an indication of the reliability of the software.

2. *Regulation of Software Creators*

Traditionally, professionals such as doctors, lawyers, pharmacists, and accountants[238] have been licensed.[239] Such licensing has been promoted as a method of protecting the health, safety, and well-being of the public by ensuring that the practitioners in the profession meet minimum standards of competence.[240] One method for as-

---

235. Gemignani, *supra* note 63, 324-25.

236. MYCIN, CADUCEUS, and CASNET are expert systems used by doctors for diagnosis and treatment of bacterial infections, internal medical problems, and glaucoma, respectively. Turley, *supra* note 111, at 456 n.9.

237. Hayes, *supra* note 3, at 115.

238. KEETON ET AL., *supra* note 81, § 32, at 185-88.

239. Graziano, *supra* note 89, at 182. *See generally* James E. O'Connor, *Computer Professionals: The Need for State Licensing*, 18 JURIMETRICS J. 256, 256-58 (1978).

240. O'Conner, *supra* note 239, at 257. O'Connor presents the following arguments in favor of licensing: (1) it meets the government responsibility for ensuring sufficient education and training for those dealing with the public; (2) it provides recourse to the public against fraud and dishonesty; (3) it encourages consistent standards by instituting penalties; and (4) it provides a central body to keep informed of scientific advances. *Id.* The arguments against licensing include: (1) it places a restriction on entrants into the field; (2) it creates a monopoly that results in increased costs; and (3) it is a government attempt to legislate morality that disrupts the free enterprise system. *Id.*

suring that the developers of expert systems possess minimum proficiency is to require that they be licensed.

At least one author has concluded that computer programmers are professionals, possessing characteristics common to a profession: Extensive training, a code of ethics imposing standards higher than normally required in the industry, disciplinary action for those breaching the code, emphasis on social responsibility rather than individual gain, and the need for a license prior to entering the profession.[241] No state, however, currently requires licensing of computer programmers.[242]

For artificial intelligence programs, several members of the development team could be licensed. First, the domain expert could be licensed to the extent possible in the particular area of professional expertise. Thus, a medical professional who provided the facts and rules for the knowledge base[243] could be licensed as a doctor or other appropriate classification by the state in which the program was written. As an alternative, national professional organizations such as the American Medical Association could certify "experts" to provide information for software development. Second, a knowledge engineer, the person who translates the rules into a computer-compatible procedure,[244] could be licensed as a computer professional by the state, as could computer programmers. Finally, similar licensing could be appropriate for other members of the development team, such as program designers and developers.[245]

If no licensing procedure for computer professionals is instituted by the state, it could be advantageous for software development companies to encourage their employees to voluntarily abide by the codes of conduct of major professional computing societies.[246]

3. *Regulation of Software Users*

Some of the same arguments that apply to the regulation of software creators also apply to the regulation of software users, especially if the user is in a position to affect the public. While it can be argued that a user should exercise care in the selection, maintenance, administration, and reliance on a particular software program and

241. MacKinnon, *supra* note 207, at 1078-80.
242. Graziano, *supra* note 89, at 182-84.
243. Reece, *supra* note 8, at 24.
244. Tuthill, *supra* note 19, at 48; Reece, *supra* note 8, at 24.
245. Reece, *supra* note 8, at 24; Tuthill, *supra* note 19, at 46-48.
246. *See* Graziano, *supra* note 89, at 183 nn. 31-32.

its results,[247] imposing such a responsibility probably will not provide adequate safeguards to the manufacturer or the user. As the user-friendliness of an expert system increasingly reflects "intelligence," it will become more difficult for a court to distinguish between a program fault and the negligence of the user.[248] While responsibility and care cannot be legislated, a requirement that software users be licensed could provide protection to the vendor and the public.

As with programmers, licensing is particularly important for people who use software affecting public health or safety or directly influencing the public well-being.[249] Expert systems providing medical information are of particular importance.[250] Similar concerns are directed to expert systems for engineering functions, architectural design, or financial planning.[251] While it may not be feasible for the user to fully verify the correctness and reliability of the software,[252] it is the user's responsibility to have a basis for deciding if the results are credible. Users should ensure first, that the software has been tested by others in the field and that it gives reliable results; second, that test programs with known solutions have been successfully run; and third, that the system has been run "in parallel" with conventional methods and has produced the same results.[253] Although users, unlike computer professionals, may not be capable of confirming the "logical correctness" of the program, they may have a duty to run a program more than once to ensure that the same results are achieved.[254] If the user were licensed and failed to perform these preliminary tests, an injured party would have additional recourse—against the user, the manufacturer, and the licensing board.

A distinction should be made between users who are amateurs and those who are professionals. For purposes of regulation, ama-

---

247. Raymond Nimmer & Patricia A. Krauthaus, *Computer Error and User Liability Risk*, 26 JURIMETRICS J. 121, 124 (1986).

248. Marshal S. Willick, *Professional Malpractice and the Unauthorized Practice of Professions: Some Legal and Ethical Aspects of the Use of Computers as Decision-Aids*, 12 RUTGERS COMPUTER & TECH. L.J. 1, 12 (1986).

249. O'Connor, *supra* note 237, at 264-65.

250. *See supra* note 235 and accompanying text.

251. *See generally* Chien & Liebowitz, *supra* note 6, at 15-18.

252. Gemignani, *supra* note 63, at 319 & n.9. This author remarks that other professionals are not required to perform extraordinary tests to ensure their "tools" are suitable for use. For instance, doctors are not required to do metallurgical testing on scalpels to ensure that blades are functional. Lawyers are not required to check all citations in a legal treatise to ensure a correct interpretation is presented. *Id.*

253. *Id.* at 320.

254. *Id.* at 322-23.

teurs are those using expert system software for their own use and without any direct interaction with the public.[255] They would not be regulated due to the difficulty of monitoring all sales and the restriction of any danger resulting from use to the user. Professionals are those using the expert system in conjunction with their occupation. This category specifically includes all those with a direct impact on public health or safety. Professionals would be regulated in order to maintain professional standards and to protect the public. Such regulations would be enforced either by the vendors, who would limit the sale of the software to those licensed to practice in the field, or by the professional organization, which would control the sale of the software to its members.[256] Such regulation alleviates problems associated with the idea that one using the software is practicing the profession without a license.[257] Regulation would not, however, address the dilemma posed when, in the professional's judgment, the proper treatment would be different from that recommended by the expert system.[258]

IV. PROPOSAL

To determine the appropriate liability standard for artificial intelligence software, both the intended function of the program and the method of selling the software must be considered. If the function is one that is potentially hazardous (e.g. engineering design, drug delivery), strict liabilty should be applied. If the intended function is nonhazardous (e.g. tax preparation, gardening advice), the method of marketing the software determines the liability standard. Strict liability should be applied if the software is mass-marketed; negligence should be applied if the software is a custom program. This proposal will allow both expert system developers and users to readily determine the appropriate liability standard, merely by considering two factors: Function and type of sale. The difficulties inherent in determining whether the end product of the software is a good or a ser-

---

255. Amateur users might be home gardeners using RootDirectory, a gardening program from GardenTech, or golfers using Personal Pro from Computer Sports. Eliot, *supra* note 5, at 13-14.

256. Medical expert systems would thus either be sold by vendors only to doctors, nurses, or other health-care professionals, or by the American Medical Association to licensed doctors.

257. Willick, *supra* note 248, at 28.

258. *Id.* at 30; *see also* Gemignani, *supra* note 63, at 325. A closely related problem is the potential for increased liability if the professional fails to use an expert system that would have predicted the proper course of action. Willick, *supra* note 248, at 8.

vice[259] would be eliminated because the emphasis would be placed on whether the software would be used for potentially hazardous activities.

To provide enhanced safety for the public, licensing requirements such as those in place for doctors, lawyers, architects, and engineers should be implemented for those who develop and use expert systems that are designed for potentially hazardous activities (e.g. medical diagnosis). The licensing will ensure that those who are in a position to create a program which could result in harm to the user will possess the requisite level of skill. This licensing can be administered either through professional organizations imposing national requirements, or by state or federal law. Domain experts, who provide the basis for the rules and facts in the expert system, should be licensed to practice in their area of expertise, just as doctors and other professionals traditionally have been. Other members of the development team—knowledge engineers and programmers—should also be licensed to ensure minimum standards of professional skill. In addition, professionals who use expert system software in their work, e.g. doctors, pharmacists, or engineers, should be certified in order to use the programs.

To ensure appropriate care in design and preparation of the software, developers of mass-marketed expert system software—the domain expert, the knowledge engineer, and the programmer, as well as the manufacturer and distributor—should all be held to a strict liability standard. For all practical purposes, such software is a "product," sold prepackaged to the buyer without any discussion or bargaining between the buyer and the developer. As such, the software should be treated as any other product in the event that harm occurs. Retailers selling mass-marketed expert systems should also be held to a strict liability standard because they are in a position that allows them to bear the costs of an injury more easily than a purchaser.[260] This standard alleviates the difficulty that a relatively unskilled user would have in proving a defect in a complicated program. Furthermore, it eliminates the need for a difficult legal analysis compounded by a complex technical discussion in a trial.[261]

Custom-written and custom-modified expert system software should be held to a negligence standard, unless an evaluation on a

---

259. *See supra* notes 114-16 and accompanying text.

260. Individual computer salespeople should not be subject to a strict liability standard. The burden should be on the plaintiffs to prove that they relied on the representations of the salesperson. Both negligence and contract remedies should be sought.

261. Turley, *supra* note 111, at 473.

case-by-case basis determines that the software is designed for use with potentially hazardous activities. For non-hazardous activities, negligence is the appropriate standard. If programs are written on a contract basis, by programmers employed for their professional skills, there is a strong indication the developers have been hired for their services and the program is not a product. Although courts have hesitated to categorize any software, including expert system software, as a service, there is some limited precedent.[262]

V. CONCLUSION

Computers and the software associated with them empower humankind to implement acts that are otherwise impossible.[263] Along with this power comes a responsibility: to use the maximum amount of care in the development and application of the programs to minimize potential risks. This obligation is particularly important in the development, sale, and use of artificial intelligence and expert system software, two types of computer programs extending their knowledge beyond that incorporated during the initial creation of the system. To provide maximum protection to the public, it is necessary to apply a standard of strict liability to any expert system that is intended for use in a hazardous activity and/or is mass-marketed. It is only by treating such an expert system as a product and not as a service that the public can be assured the chances of creating an unreasonably dangerous harm can be minimized. Such a severe standard should not, however, be applied to expert systems that are not directed to hazardous activities and which are either developed specifically for one customer or are modified, with the knowledge and/or assistance of the creator, for a customer.[264] Under these circumstances, the predominant characteristic of the programming is a service. Here, the computer developer and programmer should be treated as professionals and be held to a professional standard. In the event of a problem, a traditional negligence approach should be used. Contract remedies should also be available.

Because imposing strict liability is onerous to the developer, programmer, and vendor of the software, they might work to impose alternative standards. These include employment of licensed com-

---

262. *E.g.*, Data Processing Servs., Inc. v. L.H. Smith Oil Corp., 492 N.E.2d 314 (Ind. Ct. App. 1986). For further explanation *see supra* note 89.

263. Gemignani, *supra* note 63, at 322.

264. Modifications made without the knowledge and assistance of the creator should also be held to a negligence standard. The burden would be on the plaintiff to prove the original software developers and manufacturers were responsible for the harm.

puter professionals, certification by professional societies and trade organizations, and limitation of sales of "professional" software such as medical or legal programs to those already in possession of the requisite licenses and knowledge. Users themselves should be conscious of the potential liabilities ensuing from the use of such programs. To avoid practicing without a license, they should limit their use to areas in which they are officially sanctioned and/or experienced, and then should maintain training and updating on a regular basis.

If it is true that "[t]he computer is merely an extension of the human mind, a mere tool to enable us to expand the natural powers of the brain,"[265] it is in our best interest to continue development of software and maximize the benefits of technology. The creation and adoption of standards that address the liability issues serve two functions: First, it eliminates the uncertainties presently existing, and second, it impresses on software developers the fact they will bear the consequences of ignoring safety issues. Future artificial intelligence and expert system software will thus be as safe and effective as possible.

Marguerite E. Gerstner

265. Gemignani, *supra* note 10, at 199.