UIC John Marshall Journal of Information Technology & Privacy Law

Volume 9
Issue 4 *Computer/Law Journal - Fall 1989*

Article 5

Fall 1989

Software Vendors' Exposure to Products Liability for Computer Viruses, 9 Computer L.J. 509 (1989)

Roland B. Desilets Jr.

NOTES

SOFTWARE VENDORS' EXPOSURE TO PRODUCTS LIABILITY FOR COMPUTER VIRUSES†

I. INTRODUCTION

This Note will examine the current state of products liability law as it applies to computer software vendors focusing, in particular, on the situation of a vendor who sells a product containing a computer virus.

Computers have become a valuable, and in some cases indispensable, tool in our society. They have made it possible to store, manage, and create vast amounts of information. This was not always so. Traditionally, computers simply stored information and calculated difficult algorithms; humans were still needed to evaluate computer output and make decisions based upon that output. However, computers have gradually advanced to the point where they are capable of rendering advice and even making decisions. In addition, computers are now used to diagnose illnesses, control life support systems, manage air traffic in busy airports, control nuclear power plants, guide major weapons systems, and monitor our national defense. As computers have become increasingly responsible for managing our lives and property, there has been a growing concern about potential computer vendor liability for personal or economic damage caused by a computer system sold by the vendor.

Before examining the law of products liability as applied to the computer industry, it is necessary to understand the product itself. Computers may be thought of as consisting of two logically distinct parts: hardware and software. Hardware is the physical machine, which can range in size from hand-held to desk top to large mainframe—the large mainframe requiring a fair amount of floor space. Also, theoretically, hardware can be built to accomplish a specific purpose, to the exclusion of all others. However, to expand a computer's potential applicability, hardware is usually built in the form of a general purpose machine capable of responding to countless sets of instructions which

† This Note was awarded Honorable Mention in the Sixth Annual Computer Law Writing Competition (1989).

each yield distinct results on the same machine. The set of instructions used to accomplish a task is called software. This Note focuses on the software used to guide hardware.

Software exists for many purposes and may be obtained from several sources. The commercial types of software fall into the following four categories:

1) Software bundled with hardware to form virtually one product;
2) Software purchased as a packaged and advertised product;
3) Software purchased as a custom designed and written product; and
4) Software acquisitions that are not clearly purchases, but which are motivated by some economic transaction (e.g., a free gift from a magazine, a program copied from a bulletin board service).

It should be noted that these categories are not necessarily distinct. For instance, sometimes a standard product will be customized for a given purchaser.

This Note is primarily concerned with computer software vendors who sell a computer program outright to a purchaser. Thus, the first and fourth type of software sales mentioned will be partially excluded, i.e., sales in which software is bundled with hardware and sales in which the acquisition is not clearly a purchase.

Software that is bundled with the machine may run the machine directly for a specific purpose. An example of this is a computer monitoring program that is only sold with the machine it runs on—a machine devoted to monitoring patients. Computers of this type are not the main focus of this Note because they fall more clearly into the realm of single products (e.g., monitoring devices), rather than into the realm of separate, logical entities of software and hardware. Such packages have been treated by the courts as amounting to the sale of a "good" because of the presence of the hardware.[1] When only software is involved, the law is unclear.

Simply because the software is packaged with the hardware does not necessarily mean it will fail to be considered a separate product. For example, a computer with generalized applicability often has a program called an operating system that manages the computer's resources—its memory, disk drives, and printers—while also providing certain logical functions such as file systems, security, and the ability to initiate and terminate application programs. Although operating systems are often sold by the hardware manufacturer to the buyer when the buyer purchases a computer, in some cases, other operating systems

1. *See, e.g.*, Chatlos Systems, Inc. v. National Cash Register, Corp., 479 F. Supp. 738 (D.N.J. 1979), *aff'd*, 635 F.2d 1081 (3d Cir. 1980); Triangle Underwriters, Inc. v. Honeywell, Inc., 457 F. Supp. 765 (E.D.N.Y. 1978), *aff'd in part and rev'd in part*, 604 F.2d 737 (2d Cir. 1979) (reversed only as to the count alleging fraud in the inducement of contract, and affirmed as to all other counts), *aff'd*, 651 F.2d 132 (2d Cir. 1981).

may also be purchased separately from a different vendor. When such software can be purchased separately from the hardware, it falls squarely into the category of software as a product, even though the software is bundled with the hardware in a particular sale.

The other type of software somewhat excluded by this products liability discussion is software that is not clearly the result of a purchase. For example, bulletin boards are a common source of free programs and, as such, are a common source of viruses. It is difficult to determine if a court will apply products liability analysis to a software bulletin board operator from whose service a virus was acquired. In one respect, the bulletin board operator simply provides a means of information exchange and, thus, should not be responsible for the information placed on the bulletin board. On the other hand, perhaps bulletin boards are a product which the operator is responsible for, and which the operator should keep free from defects by inspecting items for viruses before they are posted. A plaintiff, trying to sue a bulletin board operator for damages suffered from a bulletin board virus, would need to establish that the bulletin board operator is a vendor of software before he would be able to apply the rest of the concepts discussed in this Note.

The outright sale requirement also eliminates the case where the software is leased to the customer. Courts often treat such leases as equivalent to sales.[2] This is especially true in the computer industry where companies often lease the machine to protect certain proprietary rights but are making a sale in all other respects. To apply the concepts of this Note, a plaintiff would first have to establish that the lease should be treated as the equivalent of a sale by the court.

The category that most programs fit into is application software designed to get the general purpose machine to accomplish specific goals. Examples of this type of software are word processing programs, data base management programs, and game programs. These types of programs are usually sold independent of hardware, i.e., they may be purchased without purchasing any hardware.

Having introduced the concepts of hardware and software, the concept of a software virus will now be examined. Colloquially, a computer virus is a program that, in some manner, assumes control of a portion of the computer without the rightful user's consent, often to a destructive end. In computer science terms, a software virus is a program that is dedicated to assuming control of some part of the computer without the rightful operator's consent and which can replicate and spread from one machine to another by apparently harmless contact. Examples of such contact are: (1) copying files to a machine, and (2) receiving a mail

2. *See* Rodau, *Computer Software: Does Article 2 of the Uniform Commercial Code Apply?*, 35 EMORY L.J. 853, 899-901 (1986).

message from another system. Many times the virus will remain dormant during its "incubation" period, and while dormant may be spread to other computers by users who are unaware that a virus is present. At some period in time, or after a set number of infections or other events have occurred, the program will swing into action.

The term computer virus, as currently used, encompasses all programs that act without the consent of the user, whether or not that is their sole purpose and whether or not they attempt to replicate and spread. Other computer science terms that refer to programs fitting this general description are: (1) logic bomb (a code placed within a legitimate program that will "detonate" at some later time), (2) Trojan horse (a rogue program posing as a legitimate piece of software), and (3) worm program (a program which actively tries to wriggle its way into other systems along a network).

A virus, as defined above, may take two forms: (1) it may be part of a larger, legitimate program, or (2) it may be an entity unto itself. Once it has infected a computer, a virus may do something innocent, like displaying a "peace on earth" message on the computer's monitor, or it may do something very annoying and somewhat costly, like tying up computer resources. Worse yet, a virus may do something catastrophic, such as destroying a disk full of information. Its method may be insidious, like posting random errors in an accounting spreadsheet program, or it may be blatant, like bringing the entire system to a halt. A computer virus does not care whether a computer is large or small; all computers are susceptible. While security measures may be taken to limit the likelihood of infection, any contact with the outside world is a potential exposure to a virus. Each time information enters the machine from an outside source, there is a possibility that a virus has entered as well.

Some computer virus occurrences are detailed here, since they shed additional light on the nature of viruses and their potential for damage. One such virus occurrence took place in Texas, when Donald Burleson, a programmer for an insurance company, retaliated for being fired by planting a virus program in the insurance company's computer system. The virus was activated two days after his termination. It rampaged through the system for two days before it was discovered and stopped, but not before it had destroyed over 168,000 payroll records. As a result, company paychecks were delayed for more than a month, and hundreds of thousands of dollars of further damage could have been caused had the virus not been found as quickly as it was.[3] As for Burleson, he was found guilty for violating Texas' progressive computer criminal code and will probably receive the minimum two years probation. The

3. Hafner, *Is Your Computer Secure?*, BUS. WK., Aug. 1, 1988, at 64.

Burleson case is an example of the damage a knowledgeable programmer can reek on a computer system.

Another noteworthy virus occurrence involved the Aldus Corporation. The corporation sold a drawing program called "Freehand" that contained a virus.[4] Fortunately, the virus merely caused infected computers to display the message "Peace to All Macintosh Users" for a brief period when the software was used for the first time.[5] The company recalled about 5,000 infected packages.[6] The Aldus shipment of the virus infected program exemplifies how software may become the subject of products liability litigation.

Computer viruses are dangerous. If undetected, they can become part of a product sold by a software distributor. Viruses are no longer just a series of isolated incidents. John McAfee, of the Computer Virus Industry Association in Santa Clara, California, has documented about 250,000 cases of sabotage by computer virus.[7] This Note will delineate a software distributor's liability where a virus becomes part of its product and causes serious damage to the user.

II. WARRANTY

Warranty is the first area to examine when analyzing the potential liabilities of a vendor. Specifically, the question arises as to what warranties, if any, are made or implied in the software sale. For most products, the answer to this question is found by referring to Article 2 of the Uniform Commercial Code (UCC) which addresses the rights and responsibilities of a seller of goods. For software, however, a two-step process may be required to determine which Article 2 warranties, if any, apply. This follows from the fact that software can be acquired through various methods, e.g., as a custom service, as an off-the-shelf purchase, or as part of the machine purchase, making it necessary to determine whether the software in question is a "good" under the UCC's Article 2 definition before applying Article 2 warranty law.

While it has not been absolutely determined that software is a good under the UCC, case law and most treatises on the subject lend support to the idea.[8] Cases that have dealt with the subject have involved facts in which the software and hardware were sold together. In this situation, courts have found it proper to apply the UCC to the entire transac-

4. Karon, *The Hype Behind Computer Viruses: Their Bark May be Worse than Their "Byte"*, PC WK., May 31, 1988, at 48.

5. *Id.*

6. *Id.*

7. Hafner, *supra* note 3, at 67.

8. *See infra* notes 9-10 and accompanying text.

tion.[9] While these instances do not resolve the issue of whether the UCC should apply to sales of software alone, the cases do show a willingness by the courts to apply the UCC to transactions involving software. As a consequence of such application, the possibility exists that courts will treat the sale of software as the sale of a good, and not as the performance of an intangible service.

Treatises on the subject agree that the UCC should apply, especially when the software is sold more as a "good" than as a "service."[10] For instance, when software is mass produced, advertised, and sold just like any other good, Article 2 should arguably apply. In addition, software distributors tend to frame their agreements and warranties as if the UCC did apply, so subjecting them to Article 2 would not appear to cause undue hardship.

It is important to keep in mind, however, that no court has yet ruled explicitly on this point. Meanwhile, the more the nature of the software in question gravitates away from a mass-marketed product and towards a custom service, the less likely it will be that the UCC will apply. In such a case, recovery for a damaged party under UCC warranty theory would be extremely difficult because recovery would be limited to the terms of the contract. As shall be shown, software vendors take great pains to limit their liability under their contracts.

There are basically two types of warranties under the UCC: express and implied. Express warranties are created by the behavior of the seller under section 2-313 of the UCC.[11] If the UCC applied and there

9. Chatlos Systems, Inc. v. National Cash Register, Corp., 479 F. Supp. 738 (D.N.J. 1979), *aff'd*, 635 F.2d 1081 (3d Cir. 1980); Triangle Underwriters, Inc. v. Honeywell, Inc., 457 F. Supp. 765 (E.D.N.Y. 1978), *aff'd in part and rev'd in part*, 604 F.2d 737 (2d Cir. 1979) (reversed only as to the count alleging fraud in the inducement of contract, and affirmed as to all other counts), *aff'd*, 651 F.2d 132 (2d Cir. 1981).

10. *See generally* Rodau, *supra* note 2, at 864-86; M. GEMIGNANI, COMPUTER LAW (1985).

11. U.C.C. § 2-313 (1988). Section 2-313 provides:

(1) Express warranties by the seller are created as follows:
   (a) Any affirmation of fact or promise made by the seller to the buyer which relates to the goods and becomes part of the basis of the bargain creates an express warranty that the goods shall conform to the affirmation or promise.
   (b) Any description of the goods which is made part of the basis of the bargain creates an express warranty that the goods shall conform to the description.
   (c) Any sample or model which is made part of the basis of the bargain creates an express warranty that the whole of the goods shall conform to the sample or model.

(2) It is not necessary to the creation of an express warranty that the seller use formal words such as "warrant" or "guarantee" or that he have a specific intention to make a warranty, but an affirmation merely of the value of the goods or a statement purporting to be merely the seller's opinion or commendation of the goods does not create a warranty.

was an express warranty that the product was "free from defects" or "did not carry a computer virus", this would provide an excellent basis for recovery for a purchaser harmed by a product that carried a computer virus.

However, the existence of such a warranty is unlikely. The nature of computer programs is such that vendors will not claim that the programs they sell are free from defects. However, the statement of a salesperson or representative that the product is free from a virus could form the basis of an express warranty claim. In such a situation, the court would have to determine whether the statement was an actual warranty or was simply some sort of sales "puffing" used to hype the product in order to make the sale.

Implied warranties, as delineated in section 2-314(2) of the UCC, form a better basis for recovery in this situation.[12] However, to recover under this theory, the implied warranty must, first, be found to exist and, second, must not have been successfully disclaimed by the vendor.

It is easy to argue that the software is not "fit for the ordinary purposes for which such goods are used"—that is, that it does not meet implied warranty standards—if there is a portion of viral code within the program itself that deliberately harms the consumer. Moreover, there is some question as to whether the software is "adequately contained, packaged, and labeled as the agreement may require" if it was packaged with a virus program in it. This latter point is especially compelling given the fact that, as shall be discussed below, it is possible to detect viruses through pre-packaging inspection.

Section 2-315 of the UCC speaks of the "Implied Warranty of Fitness for a Particular Purpose."[13] It is especially appropriate to apply the particular purpose warranty theory to the computer industry be-

---

Id.

12. U.C.C. § 2-314(2) (1988). The pertinent portions of the U.C.C. § 2-314(2) definition of the Implied Warranty of Merchantability are:

> (2) Goods to be merchantable must be at least such as
> (a) pass without objection in the trade under the contract description; and
> (b) in the case of fungible goods, are of fair average quality within the description; and
> (c) are fit for the ordinary purposes for which such goods are used; and
> (d) run, within the variations permitted by the agreement, of even kind, quality and quantity within each unit and among all units involved; and
> (e) are adequately contained, packaged, and labeled as the agreement may require; and
> (f) conform to the promise or affirmations of fact made on the container or label if any.

Id.

13. U.C.C. § 2-315 (1988). Section 2-315 states:

> Where the seller at the time of contracting has reason to know any particular purpose for which the goods are required and that the buyer is relying on the seller's skill or judgment to select or furnish suitable goods, there is unless ex-

cause many computer contracts are entered into with the expectation that the program will provide a "total solution" to the customer's needs. In addition, buyers frequently lack ample computer knowledge, and therefore, are likely to rely on the seller's judgment in selecting suitable goods. If the buyer has justifiably relied on the seller's expertise in choosing software, and the software proves unfit for the purpose for which it was sold because of a computer virus, recovery is likely to be allowed under the UCC.

However, under UCC section 2-316, recovery based upon these implied warranty theories can be frustrated by the use of disclaimers. Vendors typically include self-benefitting disclaimers and limitations in the software contract.[14] Courts generally accept these disclaimers if

---

cluded or modified under the next section an implied warranty that the goods shall be fit for such purpose.

Id.

14. *See, e.g.*, Gemignani, *Product Liability and Software*, 8 RUTGERS COMPUTER & TECH. L.J. 173, 177 nn.16-17 (1981) [hereinafter *Product Liability*]. Regarding disclaimers, Gemignani states that:

> A typical disclaimer reads:
>
> IBM does not make any express or implied warranties, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose.
>
> In no event will IBM be held liable for consequential damages even if IBM has been advised of the possibility of such damages.

Id. at 177 n.16 (citing R. FREED, COMPUTERS AND LAW 180 (5th ed. 1976) (quoting IBM's "Agreement for IBM Systems Engineering Services")).

Gemignani stated further that:

> Similar disclaimers appear in contracts for other IBM services, hardware and software, and in analogous contracts used by other computer vendors. Since it seems that anyone wanting to buy commercial hardware or software must accept such clauses, the clauses have the earmarks of adhesion contracts.

Id.

With regard to limitations on remedies, Gemignani noted that the following clause is standard:

> The Customer agrees that IBM's liability hereunder for damages, regardless of the form of action, shall not exceed the total amount paid for services under the applicable Service Estimate or in the authorization for the particular service if no Service Estimate is made. This shall be the customer's exclusive remedy.

Id. at 177 n.17 (citing R. FREED, *supra*, at 180).

Gemignani also stated that:

> In addition to disclaimers of warranty and limitations on remedy, computer contracts often contain strong integration clauses.

Id. (citing Schultz, *Beware of "Mined" Contracts, Attorney Warns*, COMPUTERWORLD, Jan. 28, 1980, at 15).

He then offered the following integration clause:

> THERE ARE NO UNDERSTANDINGS, AGREEMENTS, REPRESENTATIONS, OR WARRANTIES, EXPRESS OR IMPLIED (INCLUDING ANY REGARDING THE MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE), NOT SPECIFIED HEREIN, RESPECTING THIS CONTRACT OR EQUIPMENT HEREUNDER. THIS CONTRACT STATES THE

they are part of a fairly negotiated contract; if they meet the guidelines of the UCC; and if they are not against public policy.[15] These disclaimers may be voided as against public policy if they are adhesion contracts. On their face, however, these disclaimers purport to eliminate recovery based upon implied warranties.

Section 2-316(3)(b) delineates another possible defense against vendor liability. If the buyer has inspected the goods or has refused to inspect them, there is no implied warranty for defects revealed by a later examination. However, this defense, which may prove useful with regard to some software defects, will generally be inapplicable when it comes to a program virus because the purchaser will not have the expertise to detect a virus. Nevertheless, if the buyer's computer expert inspected the program for viruses, or failed to inspect for viruses after it was suggested by the seller, then a basis for eliminating all implied warranties may exist.

The vendor's ability to limit its liability under these warranties is also covered in section 2-316(4) of the UCC.[16] Computer vendors take full advantage of this subsection, and use the permissible limitation clauses to limit their liability very effectively.[17] A good example of a vendor successfully limiting its liability is found in *Chatlos Systems, Inc. v. National Cash Register Corp.*[18] Chatlos Systems, Inc. (CSI) had purchased a system from National Cash Register Corp. (NCR), but the system was never functional. CSI filed suit for breach of contract, breach of express and implied warranties, and fraudulent misrepresen-

---

ENTIRE OBLIGATION OF THE SELLER IN CONNECTION WITH THIS TRANSACTION.

Id. (quoting W.R. Weaver Co. v. Burroughs Corp., 580 S.W.2d 76, 81 (Tex. Civ. App. 1979).

15. *Product Liability*, *supra* note 14, at 178 n.19 (citing American Elec. Power Co. v. Westinghouse Corp., 418 F. Supp. 435 (S.D.N.Y. 1976) and Posttape Assoc. v. Eastman Kodak Co., 537 F.2d 751 (3d Cir. 1976) (two cases upholding these types of disclaimers)).

16. U.C.C. § 2-316(4) (1988). Section 2-316(4) stipulates that:

(4) Remedies for breach of warranty can be limited in accordance with the provisions of this Article on liquidation or limitation of damages and on contractual modification of remedy (Sections 2-718 and 2-719).

Id.

17. The following is one example of a vendor's limitation clause:

CUSTOMER'S RIGHT TO RECOVER PROPERTY DAMAGES CAUSED BY DIGITAL'S FAULT SHALL BE LIMITED TO ONE HUNDRED THOUSAND ($100,000.00) DOLLARS. DIGITAL WILL NOT BE LIABLE FOR DAMAGES RESULTING FROM LOSS OF DATA, PROFITS, USE OF PRODUCTS OR FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. This limitation of Digital's liability will apply regardless of the form of action, whether in contract or tort including negligence. Any action against Digital must be brought within eighteen (18) months after the cause of action accrues.

M. GEMIGNANI, *supra* note 10, at 112 (emphasis in original) (citation omitted). For an example of an IBM limitation clause, see *supra* note 14.

18. 479 F. Supp. 738 (D.N.J. 1979), *aff'd*, 635 F.2d 1081 (3d Cir. 1980).

tation. The trial court found for CSI, overruling NCR's argument that its obligation was "limited to correcting any error in any program as it appears within 60 days after such has been furnished."[19] NCR had also cited a "limitation of damages" clause that was included in their contract with CSI that stated: "in no event shall NCR be liable for special or consequential damages from any cause whatsoever."[20] The trial court held that both clauses were invalid as failing the essential purpose requirements of UCC section 2-719. On appeal, the Third Circuit held that, even if the exclusive remedy clause was insufficient, the disclaimer of consequential damages was still valid. The court said that clauses limiting liability are only void if they are unconscionable—a much higher standard than CSI was able to meet.

This case illustrates that recovery is possible under breach of warranty in the computer context, but that grounds for such recovery, as well as the amount recoverable, may be severely restricted by exclusionary and limitations clauses. It necessarily follows that the warranty theory of recovery will not yield much success for the consumer damaged by a software virus unless the vendor has in some way warranted the software against such an occurrence.

III. NEGLIGENCE

A software purchaser is more likely to recover damages from the vendor, due to a computer virus, if the purchaser tries to base liability on negligence. To successfully bring a negligence cause of action against a vendor, a purchaser must prove the following elements of negligence. First, the plaintiff must show that the defendant owed the plaintiff a duty of care to protect the plaintiff from injury. Second, the plaintiff must show that the defendant breached that duty by acting improperly or failing to act at all. Third, the plaintiff must show that the defendant's breach caused the plaintiff's injury. Note that this causation must exist on two levels. It must be a factual cause such that "but for" the breach by the plaintiff, the injury would not have occurred. It must also be the legal cause such that a court would find that the injury was a foreseeable result of the defendant's negligent act.[21] Finally, the plaintiff must show that he is injured. The damages recoverable under negligence include personal injuries, property damage, and, in some states, pure economic loss.[22]

The duty of care the software development industry owes to

19. *Id.* at 745 (citations omitted).

20. *Id.* (citations omitted).

21. Scott, *Negligence and Related Tort Remedies for Hardware and Software Malfunctions*, 1 COMPUTER L. PRAC. 165 (1985).

22. *Id.*

software users is not clear. Some authorities argue for establishing a professional standard and implementing the theory of computer malpractice because of the level of skill and education usually found in the programming profession.[23] They also cite the critical applications for which software is often used, and the need for expert testimony when attempting to show breach and cause in computer related cases, as reasons for setting a professional computer developer standard.[24] To further this position, it should be noted that the nature of writing custom programs is, in itself, professional, and thus calls for the setting of a professional standard.

However, there are several difficulties with the concept of computer malpractice. First, there are no minimum education or licensing requirements for becoming a software developer, nor any clearly established professional standards for practice.[25] Second, a great number of computer programs are mass-produced, mass-marketed, and are generally sold as "goods," suggesting that the selling of software deviates from the rendering of a service. Finally, no court has accepted the concept of computer malpractice, and several have expressly rejected it.[26] Thus, it does not appear that a malpractice standard will be applied in the software field.

An alternate, and arguably more plausible, standard exists for determining the duty of care owed by a software vendor. Because manufacturers must meet the "standard of reasonable care" to defeat claims for defects in design and manufacture,[27] it would seem to follow that vendors should also have to meet this standard. With this in mind, there are basically three potential sources of product defect: the design, the manufacture, and the marketing. Accordingly, proving that a defect

---

23. Nycum, *Liability for Malfunction of a Computer Program*, 7 RUTGERS J. COMPUTERS, TECH. & L. 1, 8-9 (1979).

24. *Id.*

25. *Product Liability*, *supra* note 14, at 190.

26. Chatlos Systems, Inc. v. National Cash Register, Corp., 479 F. Supp. 738 (D.N.J. 1979), *aff'd*, 635 F.2d 1081 (3d Cir. 1980); Triangle Underwriters, Inc. v. Honeywell, Inc., 457 F. Supp. 765 (E.D.N.Y. 1978), *aff'd in part and rev'd in part*, 604 F.2d 737 (2d Cir. 1979) (reversed only as to the count alleging fraud in the inducement of contract, and affirmed as to all other counts), *aff'd*, 651 F.2d 132 (2d Cir. 1981).

27. R. HURSH & H. BAILEY, 1 AMERICAN LAW OF PRODUCTS LIABILITY § 2:8 (2d ed. 1974), *quoted in* M. GEMIGNANI, *supra* note 10, at 329 n.4. Hursh and Bailey state that:

> It is but a reiteration of the rule of reasonable care to describe the duty of care as a duty to use the care, skill, and diligence in and about the process of manufacturing and preparing for market that a reasonably skillful and diligent person or a reasonably prudent person would use under the same or parallel circumstances. Thus, a manufacturer has the duty of exercising reasonable care at least to see that there is no risk of injury from negligent manufacture where the article is used in the ordinary manner for which it is intended.

Id.

was introduced through any one of these sources may be sufficient to prove a manufacturer breached its duty of care.

For purposes of proof and causation, tracing the origin of a computer defect can be extremely difficult. Not only does it require a high level of expertise to understand a given computer system, but there are many possible sources of error that may cause defects. For instance, it is often difficult to completely eliminate all possible causes. There could be a hardware error in any of the many pieces that constitute a system; there could be an error in the software program itself, or in any of the software on which it depends, such as an operating system program or database program; or, the user could have made an error when operating the system or when entering data. The net result of these possible sources of error is that the plaintiff in a negligence suit has a difficult burden of proof. Moreover, res ipsa loquitur, the theory which allows negligence to be found simply because an event has occurred, will not aid the plaintiff given that all of the treatises addressing the topic of computer negligence seem to agree that computer failure occurs too often to infer that negligence is present simply because a failure has occurred.[28]

For purposes of this Note, the reader is to assume that a computer virus has been discovered in the vendor's product; this will significantly simplify the elements of proof and causation required in the typical computer defect cases. Still, two questions are left to be answered. The first question is whether the presence of the computer virus constitutes a defect, and the second is whether the presence of a defect constitutes a breach of the distributor's duty of care.

There are three theories which may be used for proving the existence of a software defect. Under the first of these theories, if a virus were coded into an otherwise legitimate program, the plaintiff would argue that the program was mis-designed. That is, the program was supposed to accomplish certain stated goals—goals which did not include the intentional infliction of harm upon the consumer. While programs are not expected to be completely error-free, a virus can not be considered an error because it is a deviation explicitly designed into the product along the way. The errors normally expected from a program are those inadvertently, not deliberately, designed into the program.

As a defense to this mis-design theory, one might argue that the virus was not part of the design but was an unauthorized alteration of the product by someone other than the vendor, and that the vendor never intended that the product contain injurious code. This argument may well be accepted by the court. However, the problem for the vendor oc-

28. M. GEMIGNANI, *supra* note 10, at 330; Nycum, *supra* note 23, at 11; *Product Liability*, *supra* note 14, at 191.

curs when the virus was part of the product when it was sold. If the virus was part of the product as distributed, or separate from the actual program but contained in the packaging as distributed, the argument for mis-design is excellent.

As stated by Michael Gemignani:

> That there is no way to insure that testing, no matter how rigorous, will reveal all of the defects in a complex computer product is certain. This, however, does not relieve the manufacturer of all responsibility for testing. Testing and inspection must be part of the design and manufacturing processes; it is rather a question of how much inspection and testing there must be in order to avoid negligence.[29]

A second theory for proving the existence of a defect in the software is based on the idea that it is the software vendor's duty to see that the software is packaged, tested, and inspected in such a manner as to prevent the release of a product containing a virus. The existence of "vaccine" programs show that viruses can be detected in many instances. Their methods give some insight as to what might constitute reasonable precautionary steps. The methods include: scanning code files for known viruses, scanning for suspicious text in a program like "Arf, arf, GOTCHA," scanning for suspicious code such as low level disk writes, and writing over the program's stack space and any other unused portions of the media on which the program is shipped in order to protect against hidden programs.[30] It can be argued that if these are reasonable, inexpensive methods of protection for users to employ, they are also reasonable for the software vendor to perform.

A final theory for proving the existence of a software defect would focus on examining the method used to implement the program code. Before a product is marketed, work done on a software product by a main developer could be reviewed by other programmers. These other programmers could examine the software product to see that no blatant design deviations, such as viruses, are present when the product is released to the public. In addition, a thorough critique of the process used to test the program might also reveal a virus when the virus is part of the program itself.

One way to defend against a charge of negligence based on the developer's method of implementation would be to show that a thorough testing process existed and to reiterate that no amount of testing could guarantee that a virus was not present. A software distributor should

---

29. M. GEMIGNANI, *supra* note 10, at 329-30 (citation omitted).

30. Seymour & Matekin, *Confronting the Growing Threat of Harmful Computer Software Viruses*, PC MAG., June 28, 1988, at 35.

explain the reviewing process for implementation of the software and have the reviewers testify regarding the thoroughness of their review.

A software distributor should also review the safeguards on the packaging and shipping process designed to prevent a virus from being placed in the product at that stage. A distributor's failure to respond to any of these issues could result in a finding that the distributor breached its duty of care.

If the plaintiff's burden of proving duty and breach have been met, the next element of negligence is cause, both factual and legal. As stated earlier, the factual cause element can be met by showing the presence of the virus in the product as sold. By using the virus, itself, as evidence, it should be easy to show what the virus was programmed to do and what effect such action had. Consequently, proving that a defect exists when a virus is present becomes much easier than proving the existence of some other types of defects since viruses and their intended effect can often be plainly shown.

The legal causation element may be more difficult to prove. In order to recover damages from a software vendor in a negligence action, the plaintiff is usually required to prove that the damages were "reasonably foreseeable." Whether the damages are found to be reasonably foreseeable may depend on how the software was acquired, what the program was used for, and the nature of the damage sustained.

If the program comes from a custom software developer, the developer will obviously be aware of the intended use of the program. The custom developer should also be able to assess the potential damage a virus could cause; therefore, in a case such as this, foreseeability should be easily found. At the other extreme, the software developer who mass-markets a program may have no way of knowing how the program will be used or how a virus will affect consumers. For instance, a computer game containing a virus may be run on a machine that contains valuable payroll records. Accounting for the potential damage of such mass-marketed software may be very difficult indeed, making it hard to prove that particular damage was foreseeable.

The purpose for which the program is sold may also be significant in determining foreseeability. If the application of the program is known by the software vendor to involve an activity in which a software defect could cause serious consequences, the vendor may be found liable for resulting damage, if it is determined that the vendor did not use an appropriately high standard of care in the program's development.[31] For example, it is reasonably foreseeable that a small defect in a program that is used to monitor a critically ill patient or control a nuclear

31. *Product Liability, supra* note 14, at 194.

power plant could cause a great deal of damage given the program's crucial purpose.

One other factor that influences a court's determination of legal causation is the ability of the parties to bear the risk. In the software business, the ability to bear risk must be determined on a case-by-case basis. Software, even when mass-marketed, can be produced by a low-budget "garage" outfit or by such industry giants as IBM or Unisys. Similarly, software purchasers can range from a single family to one of the largest companies in existence. The nature of the vendor and the purchaser may therefore be influential in deciding who should bear the risk. Note, however, that where software viruses are present, a purchaser has the added advantage of being able to argue that he is unable to insure against such occurrences.[32]

Negligence law offers some hope of recovery for a plaintiff injured by a software virus, particularly under a theory of mis-design. If the plaintiff can show that, in the course of producing the software product, the vendor breached his duty of care by not preventing the defect, and that the resulting harm was reasonably foreseeable, the plaintiff may be successful. However, if the defendant can show that reasonable protective measures were taken to prevent such an occurrence, or that the resulting harm was not reasonably foreseeable, the defendant will be able to defeat the negligence action.

IV. STRICT LIABILITY

The final theory under which an injured plaintiff may recover is strict liability.[33] There are four primary motivations for applying this doctrine to products, all of which initially appear well suited to the computer industry. As stated by David Hall in his note on strict liability:

> First, the party in the best position to detect and eliminate defects should be responsible for damages inflicted by defective products. Second, liability should be placed upon the party best able to absorb and

---

32. Rosenberg, *Sick Software? Network Virus? Insurance Won't Help*, DATA COMM., June 1988, at 70.

33. RESTATEMENT (SECOND) OF TORTS § 402A (1965). Section 402A states:

> (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 though
> (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.

> spread the risk or cost of injuries through insurance. Third, a remedy should not be prevented by burdensome requirements of proof, since an injured person is not normally in a position to identify the cause of the defect. Fourth, due to modern marketing methods, consumers rely on the reputation of a manufacturer and no longer adhere to the doctrine of *caveat emptor*.[34]

Computer products are complicated and difficult for the average user to fully understand. As a consequence, asking a typical user to find the source and cause of a computer defect is often outside their realm of knowledge. Moreover, for proprietary reasons, most software distributors will not ship the source (human readable form of a program) of their programs to the purchaser, leaving the injured user with the even more difficult task of isolating a program defect by picking through machine code (machine readable form). It makes much more sense for the originator of a program, who possesses both the expertise and the human readable form of a program, to look for defects, such as computer viruses. It also makes sense to place the burden on the software originator to show that the software is free from the type of defect that would cause the injury claimed.

The goal of spreading risk also makes sense, although, as discussed earlier, the ability of a software distributor to bear risk may vary. Finally, many purchasers do rely on the IBM name or the Microsoft name in purchasing computer products. In these situations, the purchaser is often unable to build protections into the sales contract but must accept the terms that the vendor offers.

While the policy reasons for imposing strict liability may seem sensible, actually applying strict liability to software defects could prove extremely difficult. By examining section 402A[35] a few words at a time, the problem in applying it to the computer industry becomes apparent:

"*One who sells . . .*"[36]

This requirement of a sale seems to eliminate any "free" software copied from bulletin boards or received as gifts. It may also eliminate licensing agreements, a large segment of the industry.

". . . *any product . . .*"[37]

As discussed earlier,[38] software is frequently held to be more of a service than a product. At a minimum, custom software would seem to be exempt from strict liability. However, the more closely the software resembles a product in terms of its marketing, packaging, and volume of

34. Note, *Strict Products Liability and Computer Software: Caveat Vendor*, 4 COMPUTER/L.J. 373 (1983) (written by David Hall).

35. RESTATEMENT (SECOND) OF TORTS § 402A (1965).

36. *Id.*

37. *Id.*

38. *See supra* text accompanying notes 8-10.

sales, the more likely it is that strict liability may apply.[39]

"*. . . in a defective condition . . .*"[40]

Typically this is difficult for a plaintiff in a computer case to show with much certainty. Yet, in the case of damage from a computer virus, the defect is easily argued to be the presence of the virus.

"*. . . unreasonably dangerous . . .*"[41]

This language further limits the applicability of the rule in the computer context. Computers are frequently general purpose machines. Often, they simply provide information for people to act upon. It is difficult to argue that a generalized program is unreasonably dangerous. Yet the author of one treatise constructs a hypothetical scenario whereby a user created a recipe file using a word processor, but the program had a bug that altered one of the recipes. If the resulting meal were poisonous, the author argues that the program is unreasonably dangerous and strict liability should apply.[42] This seems to strain credulity, and it is doubtful that the courts would currently find such a product unreasonably dangerous.

It seems more likely that programs used to monitor air traffic or control nuclear power plants might be better candidates for the application of the doctrine.[43] Errors in such programs' behavior would appear unreasonably dangerous, while the presence of a computer virus in such programs could be extremely dangerous. Even if all a virus does is display a message of peace, an interruption at the wrong moment could be life threatening.

As in negligence, the plaintiff must show that the product was defective when it left the vendor's hands. Finally, the vendor is only liable under strict liability for physical harm to the consumer or to the consumer's property. As applied by most courts, this eliminates the ability to recover for economic loss.[44]

The end result is that strict liability, while not yet applied to the computer industry, could arguably be applied to a mass-marketed program with an application, which, if defective, would pose an unreasonable danger of physical harm.[45] A computer virus would be the type of defect that could, if contained in some applications, cause the risk of physical harm.

---

39. Note, *supra* note 34, at 396-97.

40. RESTATEMENT (SECOND) OF TORTS § 402A (1965).

41. *Id.*

42. Note, *supra* note 34, at 394.

43. *Product Liability*, *supra* note 14, at 197.

44. *Id.* at 197.

45. Note, *supra* note 34, at 399.

V. CONCLUSION

Pursuant to this analysis, one could conclude that there are circumstances which support recovery for the party injured by a computer virus. A cause of action based on negligence, especially where the virus was introduced through a manufacturing defect, holds the most promise for full recovery. Negligence applies regardless of the source of the software, whether custom or mass-produced. The major hurdle is proving the foreseeability of the resulting harm. This probably depends on the program's application. A vendor who sells software that monitors life support systems could foresee a great deal of harm. The vendor of a game program might foresee very little harm. The vendor of custom software has a higher degree of foreseeability since he has intimate knowledge of the software's intended use.

Strict liability may also offer a basis for recovery by a party damaged by a virus. However, the doctrine will probably only apply in the limited set of circumstances where the product is mass-marketed and the application involves a potentially dangerous activity. In addition, recovery under strict liability would be limited to physical harm to the consumer or the consumer's property.

It is important to note that there may be defenses to all of the causes of action discussed herein. Arguments such as comparative negligence, contributory negligence, express or implied assumption of the risk, misuse, and others may still be successful. This Note was meant only to set forth the possible causes of action that could be established when a software purchaser suffers injury as the result of a computer virus.

*Roland B. Desilets, Jr.**

* Mr. Desilets received his B.S. in Physics from Ursinus College in 1983, M.S. in Computer Science from Villanova University in 1988, and is currently enrolled as a third year law student in the evening program at the Widner University School of Law. He is also employed as a Senior Systems Programmer with Unisys Corporation.