UIC Law Review

Volume 30 | Issue 3 Article 9

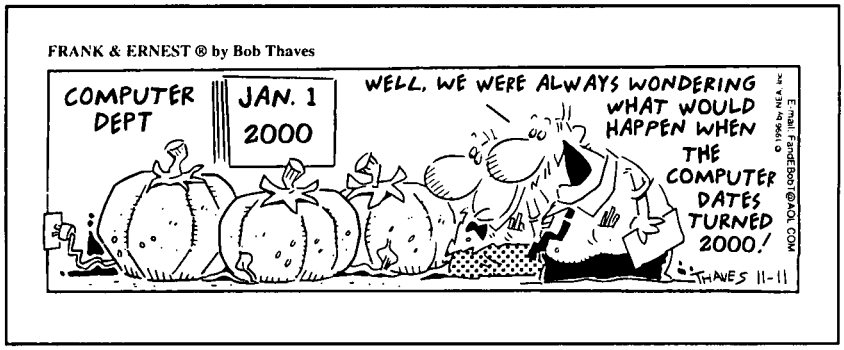
Spring 1997

Computers and the Year 2000: Are You Ready, 30 J. Marshall L. Rev. 837 (1997)

Robert G. Gerber

COMPUTERS AND THE YEAR 2000: ARE YOU READY?

ROBERT G. GERBER*



INTRODUCTION

It is December 31, 1999, the ball in Times Square is dropping and the clock is about to strike midnight. It is almost January 1, 2000. Is your computer ready? Will your computer crash or is your computer "Year 2000 compliant"?[1] The year 2000 brings with it arguably the biggest challenge the information technology ("IT") industry has ever faced.[2] The effects and costs are mind-boggling.[3]

---

* J.D. Candidate, 1998. The author would like to thank Bob Thaves for his permission in reprinting this cartoon.

1. Mary Wisniewski Holden, *Millennium Bug Threatens to Spread Mad Computer Disease*, CHI. LAW., Aug. 1996, at 78. Year 2000 incompatibility problems originated out of former computer programming practices which routinely employed two-digit date fields to represent years. *Id.* Programmers adopted this practice in order to conserve memory space on mainframe computers. *Id.* A computer is not "Year 2000 compliant" unless it recognizes four digit years. *Id.*

2. Paul Taylor, *Software Time Bomb Ticks Away: With the Year 2000 Just 47 Months Away, Many Businesses Face Serious and Costly Disruptions Because of the Way Older Computers Calculate Dates*, FIN. TIMES, Feb. 7, 1996, at I. "The year 2000 poses one of the most significant challenges ever faced by the IT industry and it will have an enormous impact on business applications, package solutions and systems software, even putting some companies out of business." *Id.* The computer's inability to perform date calculating functions is pervasive, affecting mainframes, server-based networks and personal computers. *See* Doug Bartholomew, *The Year 2000 Problem—Time is Running Out—Getting Software Ready for the Millennium Could Cost as Much as $600 Billion Worldwide. Worse, Only Half of All Companies Will be Ready*, INFO.

Year 2000 incompatibility would adversely affect routine operations such as swiping credit cards for verification,[4] setting driver's license expiration dates,[5] selling multiple year newspaper subscriptions,[6] managing air traffic control[7] and calculating insurance premiums.[8] The year 2000 will also affect the complex computer operations of the federal government.[9] Experts estimate that the

---

WK., Feb. 5, 1996, at 30-32 (discussing the enormity of the Year 2000 compliance problem, from the history of the problem to the possible solutions). *See also* INFORMATION TECHNOLOGY ASSOCIATION OF AMERICA - YEAR 2000 TASK GROUP, THE YEAR 2000 SOFTWARE CONVERSION: ISSUES AND OBSERVATIONS 1 (1996) (detailing the Year 2000 problem and the manner by which the IT industry should ensure that computer systems are Year 2000 compliant) [hereinafter ITAA, YR. 2000 SOFTWARE CONVERSION].

3. Steven Levy, *The 1,000 Year Glitch: If Computer Programmers are so Smart, How Come They Forgot About the Year 2000?*, NEWSWEEK, June 24, 1996, at 92.

4. *See It's the End of the World, and We Know It*, 5 SOFTWARE FUTURES, Jan. 1, 1996, at 3 (illustrating the Year 2000 problem in a timeline to show how the turn of the century will impact computer systems).

> Last year - though you may have been unaware of it - many five year financial forecast programs failed. This year, we'll have (at least) some five year driver license expiration problems in the US. Next year, look for insurance policy crashes; the year after, credit-card expiration snafus; the year after that, purchase order and one-year contract bugs; and the millennium will herald the start of those age calculation errors that the press will so enjoy writing [about](remember all those $10m telephone bill stories we thought we'd lived down in the Seventies?)

Id.

5. *See id.* (discussing how five-year driver's licenses will confuse computer software).

6. *See* Laurent Belsie, *Computers on Blink Over How to Read the Year 2000*, THE CHRISTIAN SCI. MONITOR, Nov. 21, 1995, at 3 (illustrating the Year 2000 problem by describing how a large New York newspaper had to stop selling ten-year subscriptions because their computers were not able to understand the end dates).

7. *See It's the End of the World, and We Know It*, *supra* note 4, at 3 (discussing how the year 2000 will affect air traffic control). In some instances, aircraft may not be able to take off because their computers will indicate that the plane has not been serviced in 90 to 100 years. *Id.*

8. *See* Pam Derringer, *Duxbury Software Firm Tackles Year 2000 Problem*, 13 MASS. HIGH TECH. 22, Oct. 16, 1995, at 1 (providing an example of how a consulting firm may face the issue of an insurance company using a computer to set insurance policy premiums). A computer that would calculate an insurance premium in 1999 would subtract the year of birth from "99" (e.g. for a person born in 1954, this equals 45). *Id.* However, in 2000 the computer will subtract "00" from "54" resulting incorrectly in 54. *Id.* This erroneous calculation would result in an incorrect insurance premium. *Id.*

9. *Computer Challenge Dateline: 1/01/00, 1996: Hearings on Year 2000 Compliance Before the Subcomm. on Gov't Mgmt., Info. and Tech. of the House Comm. on Gov't Reform and Oversight*, 104th Cong. (1996) (forthcoming 1997) (unofficial transcript at 4-5, on file with The John Marshall Law Review) (statement of George Munoz, Assistant Sec. (Mgmt.) & Chief Fin. Officer, Dep't of the Treasury) [hereinafter Munoz Testimony]. *See also* Gary H. Anthes, *Feds Face Year 2000 Crisis*, COMPUTERWORLD, Apr. 22, 1996, at 1 (discussing the federal government's estimate of $30 billion to bring the fed-

federal government and private sector businesses in the United States could spend up to $300 billion to bring the nation's computers into Year 2000 compliance, and the worldwide total could reach $600 billion.[10] This is akin to spending one dollar every second of every day for nearly 20,000 years.[11]

This Comment examines the impact of the Year 2000 on computer users and the IT industry. Part I explains the nature of the problem, explores its inception and background, and describes the parties it will affect. Part II examines the costs associated with bringing computers into Year 2000 compliance and the benefits and drawbacks of the various solutions. Part III discusses who should bear the cost of bringing the world's computers into Year 2000 compliance. Part III specifically analyzes whether software producers are liable by virtue of their warranty language for the cost of bringing their consumers' computer software into Year 2000 compliance.

I. THE YEAR 2000 PROBLEM

In approximately thirty months, computer users worldwide may face computer malfunctions or possibly the shut-down of entire computer systems.[12] Corporate executives who avoid facing and correcting their own Year 2000 problems are playing Russian Roulette.[13] Most computers programmed before 1990 recognize the year in a two-digit code.[14] The computer logic assumes that the

---

eral government's computers into Year 2000 compliance).

10. Gene Koretz, *A $600 Billion Dating Game: Why the Year 2000 Will Hurt Profits*, BUS. WK., June 17, 1996, at 30. *See also Solving the Year 2000 Software Problems, 1996: Hearings on Year 2000 Compliance Before the Subcomm. on Gov't Mgmt., Info. and Tech. of the House Comm. on Gov't Reform and Oversight*, 104th Cong. (1996) (forthcoming 1997) (unofficial transcript at 1-2, on file with The John Marshall Law Review) (statement of The Honorable Stephen Horn, Chairman, Subcomm. on Gov't Mgmt., Info. and Tech.) (discussing the costs associated with bringing the federal government and private sector computers into Year 2000 compliance).

11. *See* Bob Evans, *Year 2000 - Damocles' Sword*, INFO. WK., Feb. 5, 1996, at 6 (illustrating how enormous $600 billion is in layman's terms).

12. *See* Bartholomew, *supra* note 2, at 30-32 (warning computer users not to presume that their computer systems are Year 2000 compliant until experts prove otherwise).

13. *Id. See also* ITAA, YR. 200 SOFTWARE CONVERSION, *supra* note 2, at 2-4 (discussing the necessity for companies to recognize the problem and begin addressing and solving the problem before it is too late).

14. *See* ANDERSEN CONSULTING, TACKLING THE MILLENNIUM CRISIS 3-4 (1995) (discussing the date related programming of computers, what the Year 2000 problem is and the origins of the problem) [hereinafter TACKLING THE MILLENNIUM CRISIS]. Computers are traditionally programmed to store dates in a six-digit format —MM-DD-YY. *It's the End of the World, and We Know it*, *supra* note 4, at 3. For example, the computer stores September 16, 1996 as "09-16-96." *Id.* Hence, September 16, 2000 is stored as "09-16-00". *Id.* However, the computer understands this format as September 16, 1900 not September 16, 2000. *Id.*

first two digits of the year are "19."[15] Hence, the year 2000 presents a problem because the computer will recognize "00" as 1900 and not as 2000.[16] The effects are staggering.[17]

Processing logic that sorts data by date will be affected.[18] Programs that sort by date use logic that depends on the most recent date being the largest number.[19] The resulting problem is that "00" is smaller than "96"; therefore, the computer may place information at the wrong end of the sorted data.[20] Ironically, the Year 2000 problem is technically simple to fix.[21] However, the sheer size and scope of examining every line of code for every program in every computer system is incredibly labor-intensive.[22] Interestingly, the Year 2000 problem came about in an equally ironic manner.

This Part demonstrates the general problems related to Year 2000 compliance. Section A discusses the origins of the Year 2000 problem. Section B probes the effects of the year 2000 on the world's computer systems.

A. *Origins of the Problem*

The Year 2000 dilemma stems from two problems: two-digit year programming, and the fact that 2000 is a leap year. The most

---

15. *It's the End of the World, and We Know it*, *supra* note 4, at 3.

16. *Id.*

17. Levy, *supra* note 3, at 92.

18. C. Lawrence Meador, *Year 2000 Computers Crisis Looms for Insurers*, NAT'L UNDERWRITER, July 8, 1996, at 12.

19. *Id.*

20. *Id.*

21. *Coming Computer Problems, 1996: Hearings on Year 2000 Compliance Before the Subcomm. on Gov't Mgmt., Info. and Tech. of the House Comm. on Gov't Reform and Oversight*, 104th Cong. (1996) (forthcoming 1997) (unofficial transcript at 2, on file with The John Marshall Law Review) (statement of The Honorable Emmett Paige, Jr., Assistant Secretary of Defense (Command, Control, Comm. and Intelligence)) [hereinafter Paige Testimony].

22. *See Solving the Year 2000 Software Problems, 1996: Hearings on Year 2000 Compliance Before the Subcomm. on Gov't Mgmt., Info. and Tech. of the House Comm. on Gov't Reform and Oversight*, 104th Cong. (1996) (forthcoming 1997) (unofficial transcript at 3, on file with The John Marshall Law Review) (statement of Daniel D. Holihan, Executive Director, Data Processing Oversight Comm'n, State of Ind. on behalf of NASIRE) (discussing the number of lines of code in the States' computer systems) [hereinafter Holihan Testimony]; *The Year 2000, 1996: Hearings on Year 2000 Compliance Before the Subcomm. on Gov't Mgmt., Info. and Tech. of the House Comm. on Gov't Reform and Oversight*, 104th Cong. 7-8 (1996) (statement of D. Dean Mesterharm, Deputy Comm'r for Systems, Social Security Admin.) (discussing the number of lines of code in the Social Security Administration computer system) [hereinafter Mesterharm testimony]. The Social Security Administration has over 30 million lines of code in their computer system. *Id.* The total lines of code in each of the States ranges from 300,000 lines to over 97 million lines. Holihan testimony, *supra*, at 3.

widespread problem is two-digit year storage programming.[23] The Year 2000 date storage problem has its roots in the 1960s when computer use began to increase rapidly.[24] Mainframe systems were very expensive and rather large in size.[25] Memory space was limited and extremely expensive.[26] To conserve storage space, programmers reduced dates on documents to six digits, dramatically increasing memory and data storage space.[27] Although storage space became less expensive over time, date coding practices remained the same.[28]

In the 1970s and 1980s corporations expected their computer to be around for only a few years.[29] Corporations integrated pieces of their old computer systems into their new computer systems as they upgraded their hardware and software.[30] As a result, the old data and old systems infected the new data and new systems.[31] What saved money in the 1960s is now going to cost billions of dollars in the 1990s.[32] Software programmers did not expect these programs to be used long enough for these problems to arise.[33] In fact, it was considered arrogant by the IT industry to even suggest such a notion.[34]

The second problem arises from the fact that the year 2000 is

---

23. TACKLING THE MILLENNIUM CRISIS, *supra* note 14, at 3. This White Paper states that the most widespread programming flaw causing the Year 2000 problem is the six-digit date code. *Id.*

24. Kevin Maney, *Year 2000: Small Oversight is Big Problem*, USA TODAY, Mar. 15, 1996, at 2B; 4 WORLD BOOK ENCYCLOPEDIA 919 (1995).

25. Maney, *supra* note 24. Early mainframes were refrigerator-sized machines which frequently took up a whole room. *Id.* The speed and memory space were limited, a fraction of that available in today's PCs. *Id.*

26. *Id.*; ITAA, YR. 2000 SOFTWARE CONVERSION, *supra* note 2, at 2.

27. Maney, *supra* note 24. Coding dates YYMMDD (as compared to YYYYMMDD) greatly reduced the storage space consumed by date fields used in every document. *Id.*

28. *Id.*

29. *Id.* With the rapid increase in technology, more advanced computer systems were being developed on a regular basis. 4 WORLD BOOK ENCYCLOPEDIA 919 (1995). As a result of these new technologies, corporations only expected to have their computer systems for a few years. Holden, *supra* note 1, at 78.

30. Maney, *supra* note 24.

31. *See id.* (discussing how integrating old data into a new system can corrupt the system and result in a system that is not Year 2000 compliant). "In turn, the bug has been passed from one computer generation to another, like a faulty gene . . ." *Id.*

32. Seah Chin Siong, *Tackling the Year-2000 Digital Crisis*, BUS. TIMES, Mar. 18, 1996, at 11.

33. *See* Holden, *supra* note 1, at 78 (stating that "[m]any programmers would argue they did not use a four-digit date field because they did not expect these systems to be used long enough to cause a problem.").

34. *Id. See also* Belsie, *supra* note 6, at 3 (discussing long term use of computer programs).

a leap year.[35] Calculation of a leap year is very complex.[36] Years divisible by 100 are normally not leap years unless they are divisible by 400 as well.[37] The second millennium is evenly divisible by 100; but, it is also divisible by 400 making it a leap year.[38] Computers that treat it differently will likely malfunction.[39] Date storage and leap year logic will have a dramatic impact on computers worldwide.

B. Who and What Is Affected?

Unfortunately, no one is immune from the Year 2000 problem.[40] Affected computations include applications which calculate age, sort by date, and other date-related specialized tasks.[41] Given society's reliance on computers, the effects of a computer system failure will range from an inconvenience to enormous problems: military weapons systems failures; errors in the banking industry, specifically with ATMs and direct deposits, withdrawals and account balances; personal, medical and academic records malfunctioning; payroll and social services checks not being issued; nonis-

---

35. TACKLING THE MILLENNIUM CRISIS, *supra* note 14, at 3.

36. *Id.*

> A leap year is a year containing some intercalary period, especially a Gregorian year having a 29th day of February instead of the standard 28 days. The astronomical year, the time taken for the Earth to complete its orbit around the Sun, is about 365.242 days, or, to a first approximation 365.25 days. To account for the odd quarter day, an extra calendar day is added every four years, as was first done in 46 BC, with the establishment of the Julian calendar. Over many centuries, the difference between the approximate value of 0.25 day and the more accurate 0.242 day accumulates significantly. In the Gregorian calendar now in general use, the discrepancy is adjusted by adding the extra day to only those century years exactly divisible by 400 (e.g., 1600, 2000). For still more precise reckoning, every year evenly divisible by 4,000 (e.g., 16,000, 24,000, etc.) is made a common (not leap) year.

Id. at 3 n.1.

37. *Id.*

38. Taylor, *supra* note 2, at I. Some software is incapable of calculating the divisibility of 2000 by both 100 and 400. *Id.* Other software results in a malfunction even though it is capable of the calculation. *Id.*

39. TACKLING THE MILLENNIUM CRISIS, *supra* note 14, at 3.

40. *The Crash of "00", 1996: Hearing on Year 2000 compliance Before the Subcomm. on Gov't Mgmt., Info. and Tech. of the House Comm. on Gov't Reform and Oversight*, 104th Cong. (1996) (forthcoming 1997) (unofficial transcript at 4, on file with The John Marshall Law Review) (statement of Kevin Schick, Research Director, Gartner Group) [hereinafter Schick Testimony]. Affected applications, systems and hardware include the following: decision support systems (DSS); on-line transaction processing (OLTP); platforms; PCs; mainframes; server networks and user interfaces. *Id.* "There is a presumption that PCs and newer development languages do not have Year 2000 date problems. Wrong!" *Id.*

41. *See* ITAA, YR. 2000 SOFTWARE CONVERSION, *supra* note 2, at 2 (discussing affected computations).

suance of licenses and permits; inventory system failures; credit card verification failures; accounts receivable and payable malfunctions; and many other functions ceasing to operate or operating incorrectly.[42]

Nearly all operating systems will be affected, from older mainframes to new elaborate server-based networks to individual laptop and desk-top personal computers (PCs).[43] The effects cut across the entire IT infrastructure.[44] A significant number of applications currently in use cannot understand dates beyond December 31, 1999.[45] Over ninety percent of all Fortune 500 companies will be affected.[46]

There is a misconception that newer computer systems will not be affected.[47] When a company purchases new hardware and a new operating system, transferring old data and portions of an old system to the new hardware is dangerous.[48] The result is the contamination of the new system and new hardware with the Year 2000 "Millennium bug."[49] Nearly every company processes data in

42. *See id.* (discussing malfunctions stemming from the Year 2000 problem). *See also* Paige testimony, *supra* note 21, at 2 (discussing the effects of the Year 2000 problem on the Department of Defense).

43. TACKLING THE MILLENNIUM CRISIS, *supra* note 14, at 4.

44. Maney, *supra* note 24.

45. *See* Jeff Eckhoff, *State Vows to Avoid Computer Crash in Year 2000*, CENT. PENN BUS. J., Feb. 9, 1996, at 3 (discussing how programs are not able to understand dates beyond December 31, 1999 and that many of these programs are in use today). The following are examples of what will likely be affected worldwide when the year 2000 arrives: Programs that create and manage loan documents, insurance policies or financial projections, Siong, *supra* note 32, at 11; applications dealing with transport schedules and retirement benefits, Taylor, *supra* note 2, at I. "Every time a retailer swipes a credit card that has a 00 expiration date, the stores computers will stop the transaction, figuring that the card expired the year the first cars came with steering wheels." Maney, *supra* note 24. Loans that run beyond 2000 may be calculated improperly. *Id.* For instance, a four-year loan from 1996 through 2000 may be calculated as a negative 96-year loan, not a four-year loan. *Id. Information Week* states that the impact is particularly acute for banks and financial institutions. *See* Bartholomew, *supra* note 2, at 30 (discussing how bank systems will crash or compute incorrect information).

46. Siong, *supra* note 32, at 11 (quoting the Gartner Group estimate of the percentage of *Fortune 500* companies affected by the Year 2000 problem). "One Connecticut-based consulting firm has predicted that 90 percent of all business applications currently in use will fail—by producing inaccurate calculations, resetting dates to 1900, or simply freezing in something cybernetically like throwing a speeding car into 'park'—by the end of 1999." Eckhoff, *supra* note 45, at 3. "The year 2000 will result in 40% - 80% of all code being modified." Schick Testimony, *supra* note 40, at 7.

47. *See* Schick Testimony, *supra* note 40, at 4 (discussing the presumption that PCs and newer computer languages do not have Year 2000 problems).

48. *See* Maney, *supra* note 24 (describing the danger of transferring old data to new systems as infecting the "genes" of the new system).

49. *Id.*

a program that is date-dependent.[50] Therefore, companies ought to assume that their systems are guilty of having the Year 2000 "Millennium bug" until proven "innocent."[51] Hence, the company can allocate the necessary funds for its Year 2000 compliance projects.

II. COSTS AND METHODS OF YEAR 2000 COMPLIANCE

Experts put the price tag to bring the world's computers into Year 2000 compliance at as much as $600 billion.[52] This sum of money is 6000 times larger than the cost of a presidential campaign. It would cost less to change the 100 billion light bulbs in use worldwide.[53] Correcting software applications to recognize four-digit years is relatively simple technically.[54] The enormous cost derives from the sheer size of a line by line search to locate and modify the affected code.[55] For instance, the Social Security Administration ("SSA") currently uses over thirty million lines of code and each line needs to be examined individually.[56] The SSA estimates that approximately 300 work-years will be necessary to make and test changes.[57] Because the problem is so expensive to fix, some industry specialists predict that twenty percent of the industry's independent software vendors will go out of business.[58]

This Part addresses the costs and methods of Year 2000

50. Holden, *supra* note 1, at 78 (quoting a prominent technology attorney as stating, "[j]ust about every business has some kind of processing that depends upon date-dependent sorting or calculating. . . It's every business - it [is] just a matter of degree.").

51. *See* Bartholomew, *supra* note 2, at 30 (urging computer users to assume that their computers are not Year 2000 compliant).

52. Holden, *supra* note 1, at 78.

53. Maney, *supra* note 24.

54. *See* Eckhoff, *supra* note 45, at 3 (discussing the technology involved in correcting the Year 2000 problem).

55. Year 2000 compliance projects are so expensive because of the necessity to do a line-by-line search for affected code. *See id.* (discussing the size and cost of rewriting the world's computer code). Furthermore, as the year 2000 approaches, the modification work will become much more expensive. *See* Schick Testimony, *supra* note 40, at 8 (discussing the increase in cost each year for Year 2000 compliance projects). If a company waits until the beginning of 1999 to hire a consultant for Year 2000 compliance modification, the cost will be three times what the same project would cost today. *Id.*

56. Mesterharm Testimony, *supra* note 22, at 2.

57. *Id.*

58. Thomas Hoffman, *Small Vendors Pressed for Year 2000 Remedy*, COMPUTERWORLD, May 6, 1996, at 1. The trend for software purchasers is to put Year 2000 compliance language in all new software licenses. *See id.* (discussing the difficulty small software producers are having bringing their programs into Year 2000 compliance). Therefore, many small software producers will go out of business because they will not be able to compete with larger manufacturers who have already brought many of their programs into Year 2000 compliance. *Id.*

compliance. Section A examines the enormous cost of compliance. Section B discusses the two most common methods of Year 2000 compliance: field expansion and century derivation.

A. *Year 2000 Compliance Costs*

Many company executives believe that Year 2000 compliance expenditures have no "positive value."[59] The current industry rate for Year 2000 compliance modification ranges from about $1.50 to $2.00 per line of code.[60] With forty to eighty percent of all code affected,[61] medium to large size companies may spend, on average, $40 million apiece to bring their computers into Year 2000 compliance.[62] The United States Department of Defense ("DOD") has a unique problem facing their Year 2000 compliance project.[63] The DOD oversees more systems based on a greater variety of computer languages than any other organization in the world.[64] There are millions of lines of code undocumented or lost and in some circumstances the date functions were hard-coded in chips that are no longer in production.[65] As a result, the DOD expects to spend up to $8.52 per line of code for critical command and control software and one billion dollars overall to modify its software.[66]

States face Year 2000 problems as well.[67] States' estimates of

---

59. *See* Maney, *supra* note 24 (discussing the no "positive value" view that many companies harbor about Year 2000 compliance programs). Companies view fixing their computers Year 2000 problems as only preventing bad things from happening, not as an expenditure that is enhancing the company's computer systems with a more technologically advanced product. *Id.* It is very important for upper management to understand the problem, appreciate the affect of the problem and make the financial commitment to solving the problem within their companies. ITAA, YR. 2000 SOFTWARE CONVERSION, *supra* note 2, at 3-5. Companies can view the Year 2000 problem as a catalyst for updating their computer systems. *See* Bartholomew, *supra* note 2, at 30 (discussing how the money spent on Year 2000 compliance can have a positive value for companies).

60. James Overstreet, *Public Service Hires Firm to Help Tackle Year 2000*, THE BUS. J. - CHARLOTTE, Aug. 26, 1996, at C2. Others suggest Year 2000 compliance could cost "upwards of $2 for each line of code in fixing the problem. This could easily tally tens, or even hundreds, of millions of dollars per company." Meador, *supra* note 18, at 12.

61. Schick Testimony, *supra* note 40, at 7.

62. *Change of Centuries Promises to Wreak Havoc and Cost Billions; Not a Single U.S. Company is Year-2000 Ready, According to Information Week*, PR Newswire, Feb. 2, 1996, *available in* LEXIS, News Library, PR Newswire File [hereinafter *Change of Century*].

63. Paige Testimony, *supra* note 21.

64. *Id.*

65. *Id.*

66. Anthes, *supra* note 9, at 1.

67. *See* Holihan Testimony, *supra* note 22, at 3-4 (discussing the Year 2000 problem on behalf of the National Association of State Information Resource Executives).

lines of code range from 300,000 to over ninety-seven million lines of code.[68] Nebraska plans to spend thirty-one million dollars to bring its software into Year 2000 compliance.[69] Nebraska expects to raise its cigarette tax $0.02 per pack to pay for their Year 2000 compliance.[70]

B. Methods of Compliance

There are several ways to modify computer software to make it Year 2000 compliant.[71] Additionally, some companies may want to view the Year 2000 as a catalyst for updating and implementing an entirely new computer system.[72] Nevertheless, it is vitally important to know from where a company is beginning.[73] First, businesses should develop an inventory of all production code.[74] The second task is to scan and analyze the code and determine where date codes appear.[75] This listing will be helpful in estimating the overall Year 2000 compliance cost and the duration of the modification process.[76] The third task is the modification of the code.[77] Two possible methods of modification are field expansion and century derivation.[78] The final task is to thoroughly test all modifications before releasing a modified program for use.[79]

---

68. *Id.*

69. Koretz, *supra* note 10, at 30.

70. *Id.*

71. *See* TACKLING THE MILLENNIUM CRISIS, *supra* note 14, at 21-23 (discussing various methods of modifying code to understand the Year 2000).

72. *See* Bartholomew, *supra* note 2, at 30 (discussing how the Year 2000 problem is an opportunity for many companies to consider upgrading their computer systems).

73. *See* C. Lawrence Meador, *Technology Overview: Solving the Year 2000 Problem - Various Products and Services can Help You Plan and Manage the Project*, INFO. WK., Feb. 5, 1996, at 44 (discussing the steps to approaching the Year 2000 problem); TACKLING THE MILLENNIUM CRISIS, *supra* note 14, at 10-17 (discussing a three step approach of Assess - Plan - Implement for the Year 2000 problem).

74. Meador, *supra* note 73, at 45. "While [developing an inventory of all production code] may seem a trivial task, the importance of creating an accurate and complete listing can't be overemphasized." *Id.*

75. *Id.* "Ideally, the scanning process will produce . . . a listing of where dates are located in the source code . . ." *Id.*

76. *Id.*

77. *Id.*

78. *See* TACKLING THE MILLENNIUM CRISIS, *supra* note 14, at 21-22 (discussing the code programming for field expansion and century derivation).

79. *Id.*; Meador, *supra* note 73, at 44. The testing stage is extremely important. *Id.* The Social Security Administration will have all of its Year 2000 changes completed by December 31, 1998 in order to have an entire year to test their modifications. Mesterharm testimony, *supra* note 22, at 3. Furthermore, the Department of Treasury estimates that 45% to 55% of their total Year 2000 effort will be testing their modifications. Munoz Testimony, *supra* note 9, at 7.

1. *Field Expansion*

Determining whether expanding the date field or century derivation is an appropriate method of Year 2000 compliance can only be done after all date-related functions are identified.[80] Date field expansion will normally work, but it requires substantial testing.[81] The testing is necessary because redefinition of databases and files and new conversion routines may be needed for the modified program to function accurately.[82] Field expansion is technically rather simple.[83] A more advanced redefinition field expansion is more advantageous.[84] The redefinition expansion modifies only the affected code and leaves the remaining code unchanged.[85] Either method of field expansion will likely correct the

---

80. *See* TACKLING THE MILLENNIUM CRISIS, *supra* note 14, at 21 (discussing how date-related codes can be corrected by field expansion or century derivation).

81. *Id.* Date field expansion is a relatively simple catch-all approach. *Id.* This approach requires extensive testing to determine if further redefinition of databases and files is needed. *Id.* Additionally, this approach may require that "[s]uitable data conversion routines be built." *Id.* For a visual example of the following modification methods, consider that a program's original code looks like this:

```
01   POLICY-NO .                PIC X(10).
01   POLICY-EXPIRY-DATE-YY      PIC 99
01   WS-TODAY-YY                PIC 99

IF POLICY-EXPIRY-DATE-YY < WS-TODAY-YY
 THEN CALL 'INS0101' USING POLICY-NO.
```

Id.

82. *Id.* at 21.

83. *Id.* Field expansion requires that the programmer change two digits. *Id.* For a visual example of field expansion, consider the following modification to the original code depicted in note 81, (changes are depicted with strikethrough and underline editing):

```
01   POLICY -NO .               PIC X(10).
01   POLICY-EXPIRY-DATE-YY      PIC 99(4).
01   WS-TODAY-YY                PIC 99(4).

IF POLICY-EXPIRY-DATE-YY < WS-TODAY-YY
 THEN CALL 'INS0101' USING POLICY-NO.
```

Id.

84. *Id.* at 21

85. TACKLING THE MILLENNIUM CRISIS, *supra* note 14, at 21. This field expansion is a redefinition expansion of only the code that is affected by the year 2000. *Id.* For a visual example consider this redefinition expansion of the original code depicted in note 81; there is visibly a greater difference in the amount of modification than in field expansion as shown with strikethrough and underline editing:

```
01   POLICY -NO                 PIC X(10).
```

Year 2000 problem.[86]

2. Century Derivation

The century derivation approach corrects the Year 2000 problem by modifying the logic the computer uses in performing date-related functions.[87] The business selects a number, representing a date - "30" representing the year 2030.[88] The code is modified so that the computer reads all numbers less than "30" to be in the Twenty-first century and all numbers greater than "30" to be in the Twentieth century.[89] Therefore, century derivation is not rec-

---

01 POLICY-EXPIRY-DATE-YYYY PIC 99(4).
01 FILER REDEFINES POLICY-EXPIRY-DATE-YYYY.
05 FILER PIC 99.
05 POLICY-EXPIRY-DATE-YY PIC 99.

01 WS-TODAY-YYYY PIC 99(4).
01 FILLER REDEFINES WS-TODAY-YYYY
05 FILER PIC 99.
05 WS-TODAY-YY PIC 99.

IF POLICY-EXPIRY-DATE-YY < WS-TODAY-YY
THEN CALL 'INS0101' USING POLICY-NO.
Id.

86. "The field expansion technique will always work . . . ." *Id.* at 21.

87. Meador, *supra* note 18, at 12. Computer date sorting logic often depends on the most recent year being the largest number. *Id.* Century derivation adjusts the computer's logic to read two-digit values less a certain number as having a greater value than two-digit values greater than that number. *Id.* For example, if the programmer selected "30" (to represent the year 2030), 00 to 30 would be assigned a greater value than 31 to 99. *See* TACKLING THE MILLENNIUM CRISIS, *supra* note 14, at 22. Hence, 2000 to 2030 would be correctly viewed as more recent than 1931 to 1999 by the computer. *Id.* For a visual example of century derivation logic, suppose the following code is the original code:

01 POLICY -NO PIC X(10).
01 POLICY-EXPIRY-DATE-YY PIC 99.
01 WS-TODAY-YY PIC 99.

IF POLICY-EXPIRY-DATE-YY < WS-TODAY-YY
THEN CALL 'INS0101' USING POLICY-NO.
Id.

88. *Id.*

89. TACKLING THE MILLENNIUM CRISIS, *supra* note 14, at 22. This modification is adjusting the computers logic to read two-digit date codes as being in the correct century based on their value in relation to thirty, the selected number. *Id.* A visual example of the modified original code from note 87 will look like this, a much more complex modification of the code as strikethrough and underline editing depicts:

01 POLICY -NO PIC X(10).
01 POLICY-EXPIRY-DATE-YY PIC 99.
01 WS-TODAY-YY PIC 99.

ommended when there is information within the derivation parameters that a company desires to keep.[90] This code modification is considerably more complex than field expansion; hence, century derivation is not always appropriate.[91] When it is appropriate it is a very effective and efficient method of code modification.[92] However, once a method of modification is selected, someone must pay for the cost of the Year 2000 compliance project.

III. WHO OUGHT TO PAY FOR THE COST OF COMPLIANCE

This Part discusses whether software producers, by virtue of their warranty language, are liable for their consumer's Year 2000 compliance costs. Section A focuses on potential breach of contract causes of action. Specifically, Section A addresses several affirmative defenses such as statute of limitations, impracticability, the waste doctrine and anticipatory repudiation. Section B proposes warranty language for software producers to use in their engagement contracts prior to the turn of the century.

A. The Breach of Contract Cause of Action

This Section takes an in-depth look at a breach of contract cause of action. The section discusses the statute of limitations and explores the viability of an affirmative defense of impracticabil-

---

```
01  WS-CUT-OFF              PIC 99 VALUE 30.
01  END-OF-CENTURY          PIC 99 VALUE 99.
01  CONVERSION-YEAR         PIC 99 VALUE 95.

IF      (WS-TODAY-YY <= END-OF-CENTURY AND
         WS-TODAY-YY > CONVERSION-YEAR)
THEN
        IF POLICY-EXPIRY-DATE-YY < WS-TODAY-YY
            AND POLICY-EXPIRY-DATE-YY > WS-CUT-OFF
        THEN CALL 'INS0101' USING POLICY-NO
ELSE
        IF POLICY-EXPIRY-DATE-YY < WS-TODAY-YY
            OR POLICY-EXPIRY-DATE-YY >= WS-CUT-OFF
        THEN CALL 'INS0101' USING POLICY-NO.
```

Id.

90. *Id.* For example, if a company has data from 1925 that it wishes to preserve, than the number selected, representing the year, must be less than the year of the data desired to be preserved. *Id.* Given the example where redefined year logic reads two-digit year codes less than 30 as occurring in the 21st Century and two-digit year codes greater than 30 as in the 20th Century, data stored in the computer from 1925 (e.g. "25" which is less than "30") would either be lost or read by the computer as data from 2025. *Id.*

91. *Id.* Century derivation cannot be used when the cut-off year is too early and will not appreciably extend the life of the application. *Id.* Additionally, this modification cannot be used when the application reads dates more than a century apart, although field expansion could be used to make the application Year 2000 compliant. *Id.*

92. *Id.*

ity.[93] The Section also discusses the waste doctrine,[94] insofar as what damages would be appropriate as a remedy. Finally, the last part of this Section examines whether anticipatory repudiation is applicable,[95] and, if so, whether the consumer is duty bound to mitigate damages suffered.

This Section uses a hypothetical clause of a warranty provision to illustrate how the above referenced contract concepts interplay. Throughout the discussion of these defenses this Comment will refer to the following hypothetical warranty language: "Producer warrants that the software, which is produced to the specifications outlined above in this agreement, will run accurately on the purchaser's operating system."[96]

The warranty language prompts the question: Who ought to pay for the cost of Year 2000 compliance? Suppose a purchaser, who bought software pursuant to a contract with the above warranty, is suing the producer for breach of contract. The software producer may wish to answer the complaint with any of the following defenses.

1. *Statute of Limitations*

Arguably the statute of limitations question may be the single most important issue to address.[97] The statute of limitations is the

---

93. *See* ARTHUR L. CORBIN, CORBIN ON CONTRACTS, § 1325, at 266, n.29 (1981) (defining circumstances under which a contract may be impracticable). "A performance may be so difficult and [extremely] expensive that it is described as 'impracticable' and enforcement may be denied on the ground of impossibility." *Id.* A defense of impossibility discharges the breaching parties duty to perform. *Id.* at 265-66. "At one end of the category is the so called absolute [or "physical"] impossibility." *Id.* "The term is also used with respect to performances that are extraordinarily difficult for anybody to do and yet the experience has shown can be done by exceptionally wise and efficient men." *Id.* at 266.

94. Justice Cardozo's opinion in *Jacob & Youngs v. Kent* is the landmark case establishing the waste doctrine. "The owner is entitled to money which will permit him to complete, unless the cost of completion is grossly and unfairly out of proportion to the good to be attained." Jacob & Youngs v. Kent, 129 N.E. 889, 891 (N.Y. 1921).

95. "A repudiation is a party's manifestation that [it] is not going to provide those goods or services when they will be due at some future date." Bill's Coal v. Board of Pub. Utils., 682 F.2d 883, 886 (10th Cir. 1982) (quoting RESTATEMENT (SECOND) OF CONTRACTS § 250 (1981)).

96. In order to explore the interplay of the contract concepts discussed in this Comment, it is necessary to use a hypothetical. This hypothetical uses a warranty section of a software producers contract which may expose the producer to liability. The producer may be liable when it uses language that states that the producer warrants that the program or software will run on the operating system of the purchaser or consumer. It is in this context that Sections two, three and four of Part III discuss the contract concepts of impracticability, the waste doctrine and anticipatory repudiation.

97. Due to the fact that the statute of limitations may bar an action, it is the first substantive issue that the court will address. Upon receipt of the

first substantive issue that ought to be addressed because its result might bar a cause of action. The sale of computer software is a "transaction in goods," thereby falling within the scope of the Uniform Commercial Code ("UCC").[98] The UCC statute of limitations is four years after the action has accrued.[99] However, the parties may agree to shorten the period to not less than one year, but cannot lengthen the statutory period.[100]

The essence of this issue is when the cause of action accrues. The UCC says that the action accrues when the breach occurs, irrespective of the non-breaching party's knowledge.[101] Now, the crucial question is when does the breach occur.[102] There are many possibilities.[103] The breach may occur in the year 2000 when the software actually fails. Thus, the four years begin to run at that time, allowing plaintiffs until 2004 to bring their causes of action.

If the breach occurs at the time the product is delivered, a defendant could argue that at the time of delivery, the goods were non-conforming and therefore the software producer breached. If this is the case, then the statute of limitations would begin to run at the time of delivery.[104] Hence, when the software fails in 2000, the consumer may be barred from bringing an action against the software producer for breach of contract. Of course, this assumes that the software was delivered before 1996, four years before the software would likely fail.

The UCC states, "[a] breach of warranty occurs when tender of delivery is made, except that where a warranty explicitly extends to future performance."[105] The defendant software producer can argue that the breach of warranty, if it in fact occurred, occurred at the tender of the product.[106] However, the plaintiff will

---

plaintiff's complaint the defendant software producer will need to file an answer. The issue of the statute of limitations ought to be addressed first. It is possible, depending on how one calculates the date at which the statute begins to run, that the statute of limitations may bar an enormous number of actions.

98. *See* U.C.C. § 2-102 (1996) (defining the scope of Article 2 - Sales).

99. *Id.* § 2-725(1) (1996).

100. *Id.*

101. *Id.* § 2-725(2) (1996). "A cause of action accrues when the breach occurs, regardless of the aggrieved party's lack of knowledge of the breach." *Id.*

102. *See The Warranty Cut-Off Rule on Computer Systems*, 31 U.C.C.L.L. 2, 2-3 (1997)

103. *Id.*

104. *Id.*

> If the clock starts to run on delivery of the goods, however, it is ticking away from the very first day. With each day, the buyer is losing a day of real time. If the defect is discovered in the second year of use, the Code's four-year statute of limitations, so understood, would leave only two years of "real" time in which to act.

Id.

105. U.C.C. § 2-725 (1996).

106. *Id.* If the software producer is successful in arguing that the warranty is not one for future performance, then the breach of warranty would be con-

likely argue that the warranty was one for future performance.[107] In the hypothetical warranty language suggested above, the warranty is not one of future performance. Therefore, for warranties with similar language, a breach of warranty would occur at the time of the tendering of the product, thereby beginning the run of the four year statute of limitations.[108] Hence, if the software is tendered before 1996 the plaintiff would be barred by UCC 2-725(2).[109]

Another possible scenario is that of anticipatory repudiation. UCC 2-610 defines an anticipatory repudiation as a breach.[110] Therefore, if the software producer clearly and affirmatively repudiates its obligations under the warranty prior to the actual failure, the software producer has anticipatorily repudiated the contract and thereby breached.[111] This would mean that a cause of action accrues and the statute of limitation begins to run.[112] Moreover, if this occurs before 1996 and the consumer waits to bring a cause of action until the software fails in 2000, the statute of limitations may bar the consumer's action.

What if the software producer contacts the buyer and informs that individual or company that the purchased software will fail in the year 2000? The consumer now has notice of a future failure and this may give a defendant software producer a sound argument that a cause of action accrued at the time notice was given. If this argument prevails, then the cause of action accrued at the time notice was given.

Finally, will any breach do or must it be a material breach? What if the software fails in 2000, but the failure only affects one percent of the functioning of the software? Would this one percent

---

sidered as occurring at the time of tender of the product per U.C.C. section 2-725(2). *Id.*

107. *Id.* The software producer can argue that the warranty language states that the software will run on the consumer's operating system. This language is not an explicit warranty for future performance. Hence, when upon tender the product worked and in the Year 2000 the software fails to work, the plaintiff may not use this failure in the Year 2000 as a basis for arguing that the warranty falls into the second clause of U.C.C. section 2-725(2) concerning explicit warranties for future performance, thereby not allowing the cause of action to accrue in the year 2000. *Id.*

108. *See In re* Dynaco Corp., 200 B.R. 750, 760 (Bankr. N.H. 1996) (discussing U.C.C. section 2-725 and when the statute of limitations begins to run). The *Dynaco* Court stated that "[s]ection 2-725 should be construed to mean the physical delivery of the machine in question on the buyer's premises with acceptance by the buyer, payment therefore, regardless of any software components that may require on-site activity by the seller." *Id.*

109. *Id.*

110. *See* U.C.C. § 2-610 (1996) (stating that when a party repudiates the contract with respect to future performance, the aggrieved may treat the anticipatory repudiation as a breach and resort to any remedy for breach).

111. *Id.*

112. *See id.* § 2-725(2) (discussing when an action accrues).

breach be significant enough to trigger the running of the statute of limitations, or must the software be rendered completely non-functional for the statute of limitations to begin to run. Additionally, if the breach need only be material, does that mean that the failure need only exceed fifty percent, or must it be greater? What if the failure affects forty-five percent? Where should the line be drawn? Of course, it depends on the perspective of parties. A defendant software producer would argue that any amount of breach, be it one percent or ninety-nine percent, is enough to trigger the running of the statute. However, the plaintiff consumer will argue from the opposite side of the spectrum, namely that the breach must be material, rendering the software valueless. Presumably, this requires a far more significant percentage of failure.

The statute of limitations may bar many causes of action. Notwithstanding the above issues, many actions will prevail. If a cause of action survives this hurdle, there are several other defenses which a software producer may wish to assert. These defenses include impracticability, the waste doctrine and anticipatory repudiation.

2. *Impracticability*

An affirmative defense of impracticability is founded on the doctrine of impossibility.[113] Several degrees of impossibility exist, ranging from "absolute" or "physical" impossibility to performance that is impossible because of extraordinary difficulty.[114] These forms of impossibility are all different and distinct in their applicability in the law.[115] Included among them is impracticability.[116] The *Restatement (Second) of Contracts* states that impracticability occurs when something is so difficult and expensive it becomes commercially impracticable.[117] Additionally, an event must occur

113. CORBIN, *supra* note 93, at 266.

> [T]he doctrine of impossibility or commercial impracticability of performance as a defense to an action for breach of contract is "essentially an equitable defense," without any basis in the specific language of the contract in question or "any expression of intention by the parties," but resting firmly on the unfairness [and] unreasonableness of giving [the contract] the absolute force which its words clearly state.

Id.; Florida Power & Light v. Westinghouse, 826 F.2d 239, 263 (4th Cir. 1987).

114. CORBIN, *supra* note 93, at 265 & n.27. Taking "all human experience" into consideration, the "thing cannot be done." *Id.* Performance is impossible when it is extraordinarily difficult. *Id.* at 266. Impossibility is the "difference between 'the thing cannot be done and I cannot do it.'" *Id.* at 265 & n.27.

115. *Id.* at 266. "There will be found in common use such terms as physical impossibility, legal impossibility, impracticability, subjective and objective impossibility, personal inability, increased difficulty and frustration of object. These express varying concepts; and the applicable rules of law are not uniform." *Id.*

116. *Id.*

117. RESTATEMENT (SECOND) CONTRACTS § 261 (1981). The *Restatement*

of which the non-occurrence is a basic assumption of the contract and the party asserting impracticability must not cause the occurrence of the event.[118]

Case law has developed a three-part test for a defense of impracticability.[119] First, a contingency or event must occur.[120] Second, the risk of the contingency must not be allocated in the contract or by custom.[121] Third, the occurrence of the contingency must render performance commercially impracticable.[122] A question develops surrounding the foreseeability of the occurrence of the contingency. If the contingency is reasonably foreseeable, it must be allocated in the contract or impracticability will not apply and the duty to perform is not discharged.[123] However, foreseeability of a risk is not conclusive evidence of allocation.[124]

When an unexpected expense of performance approaches such an extreme, courts will hold that performance is a practical impossibility and discharge one's duty to perform.[125] Even when performance is not discharged, if the expense was unforeseen and performance is therefore sufficiently impeded, a new promise of performance for increased compensation has been upheld by courts.[126]

Relating to the hypothetical warranty language set forth above, the software producer has given a warranty that the pro-

---

states in relevant part:

> When, after a contract is made a party's performance is made impracticable without his fault by the occurrence of an event the nonoccurrence of which was a basic assumption on which the contract was made, his duty to render that performance is discharged, unless the language or the circumstances indicate the contrary.

Id.; *see also* CORBIN, *supra* note 93, at 266 & n.29 (discussing impracticability).

118. RESTATEMENT, *supra* note 117, § 261.

119. Transatlantic v. United States, 363 F.2d 312, 315 (D.C. Cir. 1966).

120. *Id.* The contingency is an unexpected event. Florida Power & Light v. Westinghouse, 826 F.2d 239, 263-64 (4th Cir. 1987).

121. "[T]he risk of the unexpected occurrence must not have been allocated either by agreement or by custom." *Transatlantic*, 363 F.2d at 315.

122. *Westinghouse*, 826 F.2d at 264. "[A] thing is impracticable when it can only be done at an excessive and unreasonable cost." *Id.*

123. *See Transatlantic*, 363 F.2d at 316-18 (discussing the allocation of an occurrence in the agreement).

124. "Foreseeability or even recognition of a risk does not necessarily prove its allocation. Moreover, that some abnormal risk was contemplated is probative but does not necessarily establish an allocation of the risk of the contingency which actually occurs." *Id.* at 318.

125. CORBIN, *supra* note 93, at 294-95 n.85. "[A] thing is impracticable when it can only be done at an excessive and unreasonable cost." *Transatlantic*, 363 F.2d at 315 (quoting Mineral Park Land Co. v. Howard, 159 P. 458, 460 (Cal. 1916)).

126. *See* CORBIN, *supra* note 93, at 294-95 & nn.85-87 (discussing the unforeseeability of increases and extreme costs and the breaching party's new promise of performance in exchange for increased compensation).

gram, which was produced to specification, will run on the operating system of the purchaser. A purchaser's breach of contract action might allege that the program no longer runs on the operating system and, that by virtue of the warranty language, the software producer has guaranteed both the program and the operating system. In order to address whether the software producer has an affirmative defense, this Comment looks individually at the two tests and addresses the foreseeability of the occurrence.

a. The Restatement Test

The *Restatement (Second) of Contracts* sets forth the following elements for an affirmative defense of impracticability: (1) when a contract has been made; (2) a party's performance is made impracticable; (3) due to no fault of the party's; (4) when an event occurs; (5) of which the non-occurrence was a basic assumption of the contract; and therefore, (6) the party's duty to perform is discharged.[127] This Comment breaks down the elements of impracticability and looks at each as it applies to the hypothetical warranty language.

As a preliminary matter, there must be a contract between the parties.[128] The hypothetical presupposes that the parties have contracted and signed an agreement that includes the aforementioned warranty language.[129]

The second and fourth elements need to be considered together. The second element is whether the parties' performance is made impracticable.[130] The fourth element is that an event has occurred.[131] The parties' performance is impracticable when the performance becomes impossible due to some unanticipated event.[132]

127. RESTATEMENT, *supra* note 117, § 261. The illustrations to section 261 discuss what impracticability means in the context of difficulty and expense. *See id.* at Illustration 5(d) (discussing impracticability as a theory of the defense of impossibility). "A mere change in the degree of difficulty or expense due to such causes as increased wages, prices of raw materials, or costs of construction, *unless well beyond the normal range*, does not amount to impracticability . . . ." *Id.* (emphasis added).

128. *See id.* § 261 (defining "supervening impracticability"). Discharge by supervening impracticability can only discharge one's performance when the supervening impracticability occurs after the contract is made. *Id.*

129. See *supra* note 96 for a discussion of the basis for this hypothetical.

130. *See* RESTATEMENT, *supra* note 117, § 261 (defining "supervening impracticability").

131. *Id.*

132. CORBIN, *supra* note 93, § 1325. "[When the] purpose of one of the contracting parties has been frustrated by an unanticipated event, and the court thinks that justice requires his discharge from legal duty, it has been said that performance has been made impossible by the event." *Id.* Furthermore, "[i]mpossibility, for the purpose of [discharge], means commercial impossibility. Mere increased cost of performance, *unless to an enormous and extravagant extent*, does not make it impossible." CORBIN, *supra* note 93, at 294-95 &

Additionally, as noted above, when the cost of performance is increased to an enormous or extravagant extent, the court may discharge that party's performance under this doctrine.[133] Even though the turn of the century is an expected event, many software producers did not expect that their programs, built years prior, would still be used at the turn of the century. [134] Furthermore, software producers never expected that the year 2000 would have such a debilitating effect - the malfunctioning of the delivered program - on computers.[135] It is reasonable to conclude that the malfunctioning of the delivered program, due to the unexpected continued use beyond the year 2000, is an unexpected event.[136] Furthermore, modifying the date coding to recognize four-digit years would be an enormous, excessive, and unreasonable cost of performance under the warranty.[137] Therefore, performance under the warranty is impracticable due to the unexpected event, unless the software producer is at fault.

The third element is whether the breaching party is at fault.[138] Two approaches may be taken with respect to the third element. The first is to say that the software producer has not proximately caused the software to function improperly, but rather the proximate cause of the software functioning improperly is the turn of the century.[139] Second, the issue arises whether the software pro-

---

n.85 (emphasis added). It is important to note that when the increased cost is enormous and of an extravagant nature, a party's performance may be discharged.

133. *Id.* §§ 1325, 1333.

134. Holden, *supra* note 1, at 78. Software programmers never thought that the programs they were coding would still be used at the turn of the century.

135. *See id.* (discussing why programmers did not use four-digit year codes).

136. For the purpose of finding impracticability, it is necessary to find that an unexpected event lead to performance being commercially impracticable. CORBIN, *supra* note 93, §§ 1325, 1333. Performance can be commercially impracticable when an increase in the cost of performance is so enormous that it would not be commercially practicable to perform. *Id.*; *See also Transatlantic* v. United States, 363 F.2d 312, 315 (D.C. Cir. 1966).

137. Koretz, *supra* note 10, at 30. For the software producer to perform under the warranty, the producer would have to modify the date coding of the consumer's program and operating system. Experts estimate the cost of Year 2000 compliance at $600 billion. *Id.* It has also been suggested that companies could spend up to $40 million a piece to modify their systems. *Change of Century*, *supra* note 62. In a worst case basis, if held liable, the software producer may spend $40 million or more per consumer to modify the consumer's software and operating systems. *Id.*

138. *See* RESTATEMENT, *supra* note 117, § 261 (defining "supervening impracticability").

139. This approach is based upon a literal reading of the elements. The literal cause of the malfunctioning computer is the change in the date, not the coding of the program. It is dispositive in this approach that the program has run, to specification, until the turn of century. Therefore, malfunction on January 1, 2000 suggests the turn of the century is at fault, not the software producer.

ducers are at fault because they knew that the year 2000 would have an effect on computers and yet did not change their date coding methods.[140] In the hypothetical, the question of whether the producer was at fault hinges on when the initial software was delivered to the consumer.[141] This approach necessarily asks whether the producer is at fault for not using four-digit date coding when the program was produced and delivered.[142]

Prior to 1990, the software producer ought not be at fault.[143] However, regarding contracts entered into after 1990 the answer is not as clear. The further into this decade, the more likely it is that the producer ought to be held liable for their consumers' Year 2000 compliance costs,[144] because the more likely it is that the delivered product will be used into the next century.

Either the software producer is not at fault because the software producer did not proximately cause the malfunction, or the software producer is not at fault because the program was delivered prior to 1990, and therefore, the producer and consumer did not expect the program to be used long enough for the date coding to be an issue. Alternatively, the software producer may be at fault because the program is delivered in the 1990s and, depending on the circumstances, the producer should have used four-digit date coding.

The fifth element is that the non-occurrence of the event is a basic assumption of the contract.[145] In the hypothetical, there is a basic assumption that the computer program will function.[146]

---

140. *See* Maney, *supra* note 24 (discussing the circumstances that have resulted in the Year 2000 problem and specifically the date coding practices of the programmers).

141. Time of delivery is dispositive because the programmers considered the "life expectancy" of a program as such that they did not feel that their programs would be still be in use by the Year 2000. *Id.* Computer technology is developing at such an incredible rate that the computers and software are obsolete almost as quickly as customers buy them. Holden, *supra* note 1, at 78.

142. To say that the software producer is at fault for the malfunctioning computer is to say that the software producer improperly coded the program and therefore, the turn of the century is not properly recognized.

143. This assertion is true for several reasons. First, with the rapid increase in technology, more advanced computers were being produced on a regular basis. 4 WORLD BOOK ENCYCLOPEDIA 919 (1995). Second, companies only used programs and systems for a few years. Holden, *supra* note 1, at 78. Neither the software producer nor the consumer expected that the program would be used for more than a few years. Maney, *supra* note 24, at 2B. Therefore, it would be unreasonable to hold a software producer liable for the cost of Year 2000 compliance when neither party expected that the software would be used for a long period of time.

144. Year 2000 compliance costs are the costs of modifying a computer system and all of its programs to recognize four-digit dates. Holden, *supra* note 1, at 78.

145. CORBIN, *supra* note 93, §§ 1325, 1333.

146. The computer program will function in the sense that it will operate in

Therefore, the basic assumption of the contract is the non-occurrence of the event which occurred - the malfunctioning of the software.

Under the Restatement test for impracticability, the duty to perform is discharged in two of the three distinctions drawn. If the party is not at fault because the party did not proximately cause the malfunction or because the software was delivered prior to 1990, then the duty to perform is discharged.[147] However, if the party is at fault because the software was delivered in the 1990s and the producer ought to have used four-digit year coding, then performance is not discharged and the party is liable for the Year 2000 compliance costs by virtue of the warranty language, unless the compliance is wasteful or the consumer failed to mitigate its damages after the software producer repudiated.[148] Case law has developed another approach to impracticability.

b. The *Transatlantic* Test

The second test for a defense of impracticability is derived from case law. The United States Court of Appeals for the District of Columbia established a three-prong test in the *Transatlantic Financing Corp. v. United States* opinion.[149] The *Transatlantic* Court held that: (1) a contingency or event must occur; (2) the risk of the occurrence of the contingency must not be allocated by the contract or by custom; and therefore, (3) the occurrence renders the performance commercially impracticable.[150] The court stated that the doctrine is a response to the public's interest where the commercial senselessness of performance outweighs the enforcement of the terms of the contract.[151] Performance is commercially senseless or impossible when it is not practicable, which occurs when one can only perform at an excessive and unreasonable cost.[152]

---

the manner in which it was designed. The program will perform the same operations and in the same manner on a daily basis.

147. See *supra* notes 127-46 and accompanying text for a discussion of the *Restatement* test.

148. See *infra* notes 174-94 and accompanying text for a discussion of the waste doctrine and anticipatory repudiation.

149. 363 F.2d 312, 315 (D.C. Cir. 1966).

150. *Id.*; Florida Power & Light v. Westinghouse, 826 F.2d 239, 63-64 (4th Cir. 1987).

151. *Transatlantic*, 363 F.2d at 315. The opinion states, "[t]he doctrine ultimately represents the ever-shifting line, drawn by courts hopefully responsive to commercial practices and mores, at which the community's interest in having contracts enforced according to their terms is outweighed by the commercial senselessness of requiring performance." *Id.*

152. *Id.* The court stated, "[i]t is now recognized that '[a] thing is impossible in legal contemplation when it is not practicable; and a thing is impracticable when it can only be done at an excessive and unreasonable cost.'" *Id.* (quoting Mineral Park Land Co. v. Howard, 156 P. 458, 460 (Cal. 1916)).

The Transatlantic Court did not deem excessive the additional cost of performance.[153] The Transatlantic Corporation contracted with the United States to ship wheat from Texas to Iran.[154] Transatlantic had to sail around the Cape of Good Hope due to a conflict between Israel and Egypt which resulted in the closure of the Suez Canal.[155] The court rejected the impossibility doctrine because there was a reasonable, long-used, alternate route in which Transatlantic could perform without excessive additional cost.[156] However, when there is not a foreseeable and reasonable alternative either contemplated or intended by the parties, an excessive increase in the cost of performance may excuse a party's performance.[157]

In *Westinghouse*, the court used the doctrine of commercial impracticability to discharge Westinghouse from its contractual duty to dispose of spent fuel from a nuclear plant.[158] Westinghouse

153. The Transatlantic Financing Corporation contracted with the United States for the carriage of wheat cargo from Texas to Iran. *Transatlantic*, 363 F.2d at 314. The cargo ship set sail for Iran on October 27, 1956. *Id.* On October 29, 1956, Israel invaded Egypt. *Id.* On November 2, 1956, the Egyptian government obstructed the entrance to the Suez Canal. *Id.* These events forced Transatlantic's vessel to travel around the Cape of Good Hope at an additional cost of 16% of the contract price. *Id.* at 314-15. *See also Westinghouse*, 826 F.2d at 276 (discussing the increased cost of performance in *Transatlantic*).

154. *Transatlantic*, 363 F.2d at 314.

155. *Id.* at 314-15.

156. The decision turned on the point that, although the parties contemplated the Suez Canal route, there was an alternate route, via the Cape of Good Hope, by which Transatlantic could perform without unreasonable additional cost. *Id.* at 319. The *Westinghouse* decision compared the addition performance costs borne by Transatlantic to those that would have been imposed upon Westinghouse. *Westinghouse*, 826 F.2d at 276. The court determined that Transatlantic expended an additional 16% to perform that contract, whereas Westinghouse would have had to spend and additional $80 million. *Id.* at 276-77.

157. *See generally Westinghouse*, 826 F.2d 239 (discharging performance due to an extreme increase in the cost of performance).

158. 826 F.2d 329. "Westinghouse has established its right to be excused from performance of the obligation of disposal of the spent fuel under the defense of impossibility/impracticability." *Id.* at 277. The court's reasoning aides in the use of the doctrine with respect to the Year 2000 compliance problem. Westinghouse and Florida Power & Light entered into a contract in which Westinghouse agreed to supply Florida with a PWR-type nuclear power installation and after the plant was built, but before it was operational, Florida was to choose one of three options for the disposal of the spent fuel. *Id.* at 241. The contract originated in the early 1960s. *Id.* at 240. The contract was expanded to build a second plant. *Id.* at 241. In October 1972, Florida chose an option in which Westinghouse would be responsible for the disposal of the spent fuel. *Id.* As early as 1969, Westinghouse, recognizing it may be responsible for disposal of the spent fuel, began discussions with Allied General Nuclear Services ("AGNS") to arrange for the disposal of Florida's spent fuel using AGNS's reprocessing facility. *Id.* AGNS and Westinghouse arrived at

could only perform under an alternative, namely, storage of the spent fuel, that was neither contemplated at the time the contract was entered nor intended: the parties intended disposal by way of reprocessing.[159] The uncontemplated and unintended performance by storage would have cost Westinghouse five to six times its expected profit.[160] Therefore, the court held that performance under the disposal portion of the contract was impracticable due to the excessive cost of performance.[161]

The hypothetical warranty language set forth above presents a similar situation to *Westinghouse*. The software producer intended to warrant the product and the consumer contemplated that the software producer would fix the program if it malfunctioned. The question presented is whether the parties intended that the software producer would warrant the operating system and the program years into the future. To require the software producer to perform under the warranty and reprogram both the operating system and program would be to place the software producer in a similar situation to Westinghouse. The software producer may spend forty million dollars to fix a medium to large size company's operating system, an amount that likely exceeds the bargained-for consideration.[162] Therefore, under *Westinghouse*, software producers have a potentially successful impracticability defense if they can show the following: (1) the kind of performance both parties contemplated and intended in the warranty; but (2)

---

an agreement in 1974, but AGNS refused to sign the agreement claiming that government regulations changed and the agreement would need to be renegotiated. *Id.* at 241-42. In mid-1975, Florida contacted Westinghouse to remove the cooled spent fuel from its plant, Westinghouse refused and Florida sued for specific performance. *Id.* at 242. The court found that "it was a basic assumption of both parties . . . that the disposal of the spent fuel would be effected by reprocessing, . . . that they based the bargained-for-price to be paid Westinghouse on the basis of the cost of reprocessing." *Id.* at 271. Furthermore, the court determined that reprocessing was an available and practicable method of disposal. *Id.* It was the method intended by the parties upon Florida's selection of the option in which Westinghouse was responsible for disposal. *Id.* Additionally, if the reprocessing was not available in reasonable terms, the government would reprocess the plant's spent fuel. *Id.* The court also found that the only available method of disposal was storage, at a cost of $80 million, which would completely wipe out the bargained-for profit of $18 million. *Id.* at 277. The court further reasoned, unlike in *Transatlantic*, there was no reasonable alternate performance. *Id.* The alternate course of performance would have imposed an excessive additional cost and therefore, Westinghouse established a defense of impracticability. *Id.*

159. Unlike the facts in *Transatlantic*, the storage of the spent fuel was neither contemplated by the State of Florida or Westinghouse nor did either intend for the spent fuel to be disposed of in that manner. *Westinghouse*, 826 F.2d at 276.

160. *Id.*

161. *Id.* at 277.

162. *See Change of Century*, *supra* note 62 (discussing the cost per company for Year 2000 compliance projects).

that the kind of performance demanded now was neither contemplated nor intended at the time the parties contracted; and (3) that the cost of performance is so excessive that not only is the bargained-for-profit eliminated, but it would cost software producers millions of additional dollars to perform.[163] It is vitally important to the doctrine of impracticability that the parties allocate foreseeable occurrences in the contract.

c. The Foreseeability of the Occurrence Test

If an occurrence is foreseeable, the contract must address, or courts may infer that the risk is assumed.[164] This follows from the elements found both in *Transatlantic* and the *Restatement* that the occurrence of the event be unexpected.[165] Although foreseeability is relevant, it is not dispositive.[166] Even if an abnormal risk is contemplated, it does not establish an assumption of the risk.[167] In *Transatlantic*, the Transatlantic Corporation displayed a willingness to assume an abnormal risk due to the circumstances surrounding the Suez Canal at the time they entered the contract with the United States.[168]

The situation the hypothetical warranty language presents is distinguishable from the circumstances in *Transatlantic*.[169] If a

163. *See Westinghouse*, 826 F.2d at 276-77 (discharging the duty of performance due to impracticability).

164. United States v. Winstar Corp., 116 S. Ct. 2432, 2469-70 (1996) (quoting Lloyd v. Murphy, 153 P.2d 47, 50 (Cal. 1944)). "[A]s Justice Traynor said, '[i]f [the risk] was foreseeable there should have been provision for it in the contract, and the absence of such a provision gives rise to the inference that the risk was assumed'" *Id.* The premise was that the parties ought to bargain with respect to risks that are material to the contract. *Id.* at 2469.

165. *See* Transatlantic v. United States, 363 F.2d 312, 315 (D.C. Cir. 1966) (discussing the elements of impracticability); RESTATEMENT, *supra* note 117, § 261 (discussing the elements of impracticability).

166. *Winstar Corp.*, 116 S. Ct. at 2470 n.53 (quoting 2 E. ALLEN FARNSWORTH, FARNSWORTH ON CONTRACTS § 9.6, at 555-556 (1990)).

167. The fact that the abnormal risk was contemplated may be probative. *Transatlantic*, 363 F.2d at 318. In *Transatlantic*, the court found that the circumstances surrounding the Suez Canal indicated a willingness of Transatlantic to assume abnormal risks. *Id.*

168. *Id.* It is important to note that at the time Transatlantic entered into the contract with the United States, the Israeli-Egyptian conflict had begun, and two days after the Transatlantic vessel set sail for Iran the Suez Canal was blocked. *Id.* at 314.

169. In the hypothetical, the abnormal risk is the warranty that the operating system is Year 2000 compliant when the software producer has no knowledge of the operating system or how the year 2000 will affect the system. Additionally, the warranty is not contemplated nor intended by the parties. Also, in many cases the warranty was entered into many years before the year 2000 and the software producers never even contemplated that the programs would still be in use in the year 2000. *See* Holden, *supra* note 1, at 78 (noting that software programmers did not intend their programs to be used in the year 2000).

court finds that the risk of malfunction in the year 2000 is a foreseeable risk, there is only an inference that the risk is assumed.[170] Additionally, the determination of whether the risk is foreseeable turns on the date the contract was entered into and when the product was delivered.[171] Producers of software products delivered prior to 1990 were not likely to have foreseen a year 2000 malfunction risk, while producers of products delivered after 1990 may have, depending on the intended use of the product.[172] Moreover, a program intended as a legacy program,[173] and delivered after 1990, may foreseeably be used well into the Twenty-first Century; therefore, the software producer may be held as assuming the risk by not allocating for the risk in the contract. On the other hand, an impracticability defense may be moot because to require the software producer to pay sums that far exceed the value of the program may be wasteful.

3. The Waste Doctrine

The waste doctrine states that a party should receive the cost of completion of the contract, unless the cost is grossly and unfairly disproportionate to the good attained by completion of the contract.[174] When a cost of completion award is wasteful, the remedy is the diminution in value.[175] The diminution in value is the market value of the product as specified in the contract less the

170. *Winstar Corp.*, 116 S. Ct. at 2469-70.

171. See *supra* note 143 and accompanying text for a discussion of why programs that were contracted for and delivered prior to 1990 should be considered as a separate category of programs.

172. Programs delivered after 1990 have different issues and problems. Producers having programs delivered prior to 1990 would not have had a need to allocate the risk of malfunction in 2000 because it would not be contemplated that the programs would still be used in 2000. *See* Holden, *supra* note 1, at 78 (discussing some programmers' beliefs as to longevity of their products). However, programs delivered after 1990, depending on the program and its intended benefit and use to the consumer, may or may not have been foreseen as lasting into the year 2000 and beyond. It is foreseeable that critical function programs (legacy programs) are used longer than incidental programs. Therefore, the date the contract is entered into may be dispositive depending on the intended use of the software.

173. Legacy programs are intended for long term use and are usually critical function programs. *See* ITAA, YR. 2000 SOFTWARE CONVERSION, *supra* note 2, at 4 (discussing legacy programs and why it is dangerous and irresponsible to ignore the Year 2000 problem).

174. The New York Court of Appeals, in an opinion written by Justice Cardozo, established the waste doctrine which states, "[t]he [party] is entitled to the money which will permit him to complete performance, unless the cost of completion is grossly and unfairly out of proportion to the good attained. When that is true, the measure is the difference in value." Jacob & Youngs v. Kent, 129 N.E. 889, 891 (N.Y. 1921).

175. *Id.*

value of the product as delivered.[176]

An example of the application of the waste doctrine is found in the New York case of *Jacob & Youngs v. Kent*. The New York Court of Appeals found it wasteful for the contractor to demolish a substantially completed home to replace the plumbing pipes when the replacement of the pipes would have a nominal effect if any at all, on the market value of the home.[177] The court held that a proper remedy was the diminution in value of the home, which was nominal.[178]

Under the language of the hypothetical warranty, as provided earlier in this Section, the cost of completion would be the cost of modifying both the program and the consumer's operating system.[179] This could cost the software producer nearly forty million dollars for medium to larger corporate consumers and several million dollars for smaller corporate consumers.[180]

The question turns on whether the cost of completion or performance under the warranty is grossly and unfairly out of proportion to the good attained by the consumer.[181] Again, as with impracticability, the answer depends on when the contract and warranty were entered into and when the product was delivered.[182] A company that has benefited from the utility of a program for a decade or more has realized the good attained by the contract. To require a software producer to pay potentially the contract price or more to modify a program that is technologically obsolete would be wasteful. The cost of performance under the warranty would grossly and unfairly outweigh the good attained by the consumer.[183]

---

176. Rivers v. Deane, 619 N.Y.S.2d 419, 420 (N.Y. App. Div. 1994).

177. *Jacob & Youngs*, 129 N.E. at 891.

178. *Id.*

179. See *supra* note 96 and accompanying text for a discussion about the hypothetical warranty language.

180. For a discussion of the cost of Year 2000 compliance see Part II.A of this Comment. *See* Holden, *supra* note 1, at 78 (discussing the cost of Year 2000 compliance); ITAA, YR. 2000 SOFTWARE CONVERSION, *supra* note 2, at 2-3 (discussing the cost of Year 2000 compliance).

181. *See Jacob & Youngs*, 129 N.E. at 891 (defining the waste doctrine).

182. See *supra* notes 141-42 and accompanying text for a discussion of the impact the date of delivery and the date the contract was entered into has on the issue of the software producers liability.

183. It would be difficult to say that the consumer had not already realized the bargained-for good of the product. To hold that the software producer must pay the cost of reprogramming a product that is likely obsolete in technological terms is wasteful. It would probably cost the software producer much more than the contract price to bring the program and system into Year 2000 compliance. This, like in *Jacob & Youngs*, would be allowing a measure of damages that would be grossly out of proportion to the market value of the product. *See Jacob & Youngs*, 129 N.E. at 891 (establishing the waste doctrine). There is a distinction being drawn between the value of the product to the consumer and the market value of the product. The consumer may value

If the product was delivered after 1990 however, the question then turns on what was the intended use of the program. Here, as with impracticability, if the program is a legacy program, the remedy may be the cost of completion of the contract.[184] However, if the program is not a legacy program, then the remedy ought to be the diminution in value, which may be nominal.[185] Furthermore, the software producer ought not be liable for costs that the consumer should have avoided once the software producer indicated its intention not to perform under the warranty.

4. Anticipatory Repudiation

A repudiation is an affirmative indication that a party will not perform or complete performance under the contract.[186] A mere expression of doubt as to performance is not enough to constitute a repudiation.[187] The *Uniform Commercial Code* takes repudiation one step further in the doctrine of anticipatory repudiation.[188] When a party repudiates a contract with respect to future performance, the non-repudiating party may await performance for a "commercially reasonable" time or treat the repudiation as a breach and seek redress; and the non-repudiating party must sus-

---

the product at a level that far exceeds the market value. If in fact this is the case, then the consumer may be willing to pay for the additional cost of the Year 2000 modification. In this situation, the software producer could promise to perform under the warranty for increased compensation. *See* CORBIN, *supra* note 117, at 294-95 & n.85 (discussing the breaching party's promise of performance for increased compensation).

184. See *supra* note 173 for a discussion of legacy programs. A legacy program delivered after 1990 arguably is intended to last at least a decade in order for the consumer to realize the bargained-for benefit of the software.

185. Just as the New York Court of Appeals held that it was wasteful to replace all the plumbing pipes in a nearly complete home, it would be wasteful to modify a non-legacy, technologically obsolete program. *See Jacob & Youngs*, 129 N.E. at 891 (establishing the waste doctrine). The appropriate measure of damages is the diminution in the value of the software. *Id.*

186. RESTATEMENT, *supra* note 117, § 250. This section defines a repudiation in the following manner:

> A repudiation is
>
> (a) a statement by the obligor to the obligee indicating that the obligor will commit a breach that would of itself give the obligee a claim for damages for total breach under § 243, or
>
> (b) a voluntary affirmative act which renders the obligor unable or apparently unable to perform without such a breach.

Id. "In order to constitute a repudiation, a party's language must be sufficiently positive to be reasonably interpreted to mean that the party will not or cannot perform." *Id.* at § 250 cmt. b.

187. *Id.* "Mere expression of doubt as to his willingness or ability to perform is not enough to constitute a repudiation, although such an expression may give an obligee reasonable grounds to believe that the obligor will commit a serious breach and may ultimately result in a repudiation under the rule." *Id.*

188. *See* U.C.C. § 2-610 (1996) (defining anticipatory repudiation).

pend performance and avoid further damages.[189] Therefore, the non-repudiating party has a duty to mitigate its damages.[190] In other words, damages suffered while the non-repudiating party awaits performance are not recoverable if the party waits beyond a commercially reasonable period of time.[191]

The hypothetical warranty language stated above in Part III.A of this Comment sets up a situation in which the consumer is invoking the warranty provision of the contract, claiming that the software producer warrants the delivered program and the operating system on which it runs. [192] If the software producer displays an affirmative indication or expressly states that it has no intention of performing under the warranty, the manifestation would be an anticipatory repudiation.[193] The consumer may bring a cause of action for breach of contract, or can await performance for a commercially reasonable period of time; but the consumer must mitigate the damages suffered.[194]

Questions arise as to what are the damages, when are they suffered and how the consumer can mitigate.[195] The damage suf-

189. *Id.* Section 2-610 provides that the non-repudiating party may (1) await performance for a "commercially reasonable" time; or (2) seek redress under a breach of contract cause of action; and (3) in either case, the non-repudiating party has a duty to mitigate the damages suffered by the repudiation. *Id.*

190. "[I]f [a party] awaits performance beyond a commercially reasonable time he cannot recover resulting damages which [the party] should have avoided." *Id.* at § 2-610 cmt. 1.

191. *Id.* The *Uniform Commercial Code* defines reasonable time as any time which is not manifestly unreasonable and that depends on the nature, purpose and circumstances of such an action. U.C.C. § 1-204 (1996)

192. See *supra* note 96 and the accompanying text for a discussion of the utility of the hypothetical.

193. *See* U.C.C. § 2-610 (1996) (discussing the doctrine of anticipatory repudiation); RESTATEMENT, *supra* note 117, § 250 (defining a repudiation).

194. The *Uniform Commercial Code* states that the aggrieved party must cease performance and avoid increasing the damages suffered pursuant to section 2-704. U.C.C. § 2-610(c). Section 2-704 states that the party must mitigate by completing performance and selling goods to another or discontinue performance and sell the goods for scrap. U.C.C. § 2-704(2).

195. Presumably the damages suffered will not begin until 2000. Therefore, from 1997 until 2000 the consumer may not suffer any damages. This leaves the consumer with two and a half years in which to take affirmative steps to avoid damages. Surely, the commercially reasonable time will have long passed when the year 2000 arrives. Therefore, it would seem, damages suffered after the year 2000 will not be recoverable due to the doctrine of anticipatory repudiation. See *supra* notes 186-87 and accompanying text for a discussion of anticipatory repudiation. Hence, if the consumer does nothing, then the consumer has not mitigated his damages and ought not recover for the damages suffered when its computer malfunctions in 2000. The question remains what can the consumer do to mitigate? During the two and a half years prior to the year 2000, the consumer could purchase a new program, modify the existing program, or replace the operating system which will result in the damage to be suffered in 2000.

fered by the consumer would be the malfunction of the computer program and possibly the entire computer system.[196] The damages are not suffered until the year 2000.[197] Hence, the consumer would have over two years to mitigate, and it would not be commercially reasonable for the consumer to await performance for two and a half years.[198] The consumer's mitigation may consist of purchasing a new program or system, modifying the existing program or system or just replacing the operating system that the software producer did not supply to the consumer.[199]

The software producer may be liable for the balance of the damages suffered, beyond those which were avoided by the consumer. Alternatively, the software producer's duty to perform under the warranty may be discharged due to the doctrines of impracticability or waste. A discharge due to impracticability or waste turns on the question of when the parties entered into the contract and when the product was delivered to the consumer. In the alternative, the consumer could bring a cause of action under another legal theory, such as negligence.[200] Finally, the software producer, when entering into contracts today, should choose warranty language which accurately expresses its intentions, thereby avoiding some of the liability issues this Comment discusses.

B. Proposed Warranty Language

This Section proposes warranty language designed to protect software producers from liability to their consumers for the consumer's Year 2000 compliance costs. As discussed above, certain contract language may suggest that the software producer is warranting that the consumer's operating system is Year 2000 compli-

196. See *supra* note 195 for a discussion of the damages.

197. See *supra* note 195 for a discussion of when the damages are suffered.

198. See *supra* note 195 for a discussion of the time between the repudiation and the damages suffered. *See* U.C.C. § 1-204(1) (defining reasonable time as that which is not manifestly unreasonable).

199. See *supra* note 195 for a discussion of possible mitigation and avoidance measures.

200. A negligence cause of action requires that the plaintiff plead and prove the following: (1) the defendant (2) had a duty (3) that was breached or that the defendant acted carelessly and (4) that the breach or careless act proximately caused (5) an injury to the plaintiff. *See* W. PAGE KEETON ET AL., PROSSER & KEETON ON THE LAW OF TORTS § 30, at 164 (5th ed. 1984) (discussing the elements of a negligence cause of action). Although a negligence cause of action may lie irrespective of the contact language in any particular engagement, the focus of this Comment is on a contract remedy, pursuant to the warranty language used by the software producer. For a thorough discussion of a negligence cause of action see *id.* §§ 28-45, at 160-321; Gregory C. Keating, *Reasonableness and Rationality in Negligence Theory*, 48 STAN. L. REV. 311, 325-60 (1996); *see generally* Gary T. Schwartz, *The Myth of the Ford Pinto Case*, 43 RUTGERS L. REV. 1013 (1991) (discussing foreseeability in negligence and the Ford Pinto case).

ant. The software producer who desires to avoid this liability needs to choose its contract language, more specifically its warranty language, carefully. The software producer who desires to avoid such liability needs to use language that illustrates what the producer intends to warrant. Software producers should warrant that their programs are Year 2000 compliant, but should not warrant that the consumers operating system is Year 2000 compliant.[201] The Sub-section below sets forth recommended Year 2000 warranty language. Sub-section 2 defines a few crucial terms used in the proposed language.

1. Proposed Language

The following proposed Year 2000 warranty language protects the consumer and states exactly what the software producer intends to warrant:

> Contractor warrants that the software, which is produced to specifications outlined above in this agreement, to be used by purchaser prior to, during or after the calendar year 2000, includes or shall include design and performance so the purchaser shall not experience software abnormally ending and/or invalid and/or incorrect results from the software in the operation of the business of the purchaser. The software design to ensure Year 2000 compatibility or compliance shall include, date data century recognition, calculations that accommodate same century and multi-century formulas and date values and date data interface values that reflect the century. This software will accurately function using valid dates.

This language warrants that the software program is Year 2000 compliant and neither mentions nor alludes to any Year 2000 warranty for the operating system.

2. Definitions of Terms

The following suggested definitions will be helpful in interpreting the proposed warranty language. If the producer warrants that the software will "accurately" operate in the Twenty-first Century, this term should be defined similar to the following: "Accurately" refers to correct processing according to the following criteria: calculations must execute using dates with four-digit year logic, interfaces and reports - supplied by purchaser - must support four-digit year processing, correct results in forward and backward data calculation spanning century boundaries must be provided by purchaser, including the conversion of previous years currently stored as two-digits.[202] Additionally, when referring to a "valid

201. The hypothetical warranty language as stated in Part III.A is an example of language which potentially extends a warranty to the consumers' operating system.

202. It is important to define "accurately" and be clear that it is the con-

date," the producer may want to define the term similar to the following: A "valid date" is a date which contains a two-digit month, a two-digit day and a four-digit year.[203]

The proposed language and definitions are designed to express the manifest intent of the software producer and consumer. This language warrants to the consumer that the software is Year 2000 compliant and that the consumer has certain responsibilities with respect to the compliance of the data that they import into the new system.

CONCLUSION

The year 2000 presents one of the biggest challenges the IT industry has ever faced. The problem arises due to the computers' inability to recognize four-digit years. The date-coding practices of the 1960s and 1970s will cost computer users worldwide potentially as much as $600 billion.

The crux of the Year 2000 problem is who ought to pay for the Year 2000 compliance projects. Many software producers may find themselves in breach of contract suits arising out of the Year 2000 compliance costs. Software producers warranty language could be construed as extending a warranty to the operating systems on which their delivered program runs. Software producers may avoid suits by asserting that the statute of limitations has run. This argument hinges on the defendant's ability to argue the earliest date possible for triggering the statute of limitations. The software producer may be able to avoid performance under the warranty using the defense of impracticability. If a program was delivered prior to 1990, the software producer's duty ought to be discharged.[204] However, the software producer may be liable if the program was delivered after 1990 and the program is a legacy program.

The waste doctrine may discharge performance.[205] It may be that it is wasteful to spend the contract price or more to fix a program that is technologically obsolete. Finally, if the software producer repudiates the warranty, the consumer has a duty to mitigate its damages suffered beyond a commercially reasonable

---

sumer's responsibility to see that the old data, which will be imported into the new program, is Year 2000 compliant. Therefore, if the program malfunctions in 2000, and if the cause of the malfunction is the old data, the software producer would not be responsible for the needed Year 2000 compliance modification.

203. *See* Maney, *supra* note 24 (discussing how the integration of old data into a new system can corrupt the new system). Again, it is important that the consumer knows that data without "valid dates" will adversely affect the new program in a potentially catastrophic manner. *Id.*

204. See *supra* Part III.A.2 for a discussion of impracticability and why the date of delivery may be a dispositive fact in determining liability.

205. See *supra* Part III.A.3 for a discussion of the waste doctrine.

time.[206] Since damages will not be suffered until the year 2000, it stands to reason that nearly all of the damages may be avoided.

Finally, it is critical that software producers accurately express their warranty intentions in their contracts. The warranty language, proposed above, is written with that objective as its goal.

206. See *supra* Part III.A.4 for a discussion of anticipatory repudiation and the non-repudiating party's duty to mitigate damages after a commercially reasonable period of time.