

6

Software Ownership and Intellectual Property Issues

The case studies in this chapter concern the moral question of software ownership and the scope of intellectual property rights. During the last decade or so there has been a steady erosion of respect for intellectual property laws. Many end users and professionals seem to have few qualms about making illegal copies of commercial software, and several studies confirm that such unauthorized copying is widespread.¹ In addition, in a recent *Computerworld* poll, 53 percent of the information professionals surveyed indicated that they had made unauthorized copies of software programs.² Most of these individuals rationalized their behavior by saying that they were just "trying out" the software.

Of course, intellectual property issues are more complex than pilfered copies of software and, given that complexity, it is instructive to consider briefly the whole conception of property and its subset, intellectual property. Most contemporary analyses equate the notions of "ownership" and "property." Hence, the statements, "I own that house," and "that house is my property," are equivalent, since they convey the same information. Further, those analyses define ownership as "the greatest possible interest in a thing which a mature system of law recognizes."³ Simply, ownership of property implies that the owner has cer-

tain rights and liabilities with respect to this property, including the rights to use, manage, possess, exclude, and derive income.

Intellectual property rights imply that someone has the right to certain concepts, knowledge, or ideas. However, as we observed in Chapter 1, there are serious difficulties with the notion that one has property rights in an idea, since this would mean the “right” to exclude others from using and building on those ideas. Rather, it is generally recognized that one has property rights in the way one expresses ideas, whether it be in the form of a book, a play, or a software program. Thus, the distinction between an idea and its expression serves as a fundamental basis for making legal and moral decisions about intellectual property protection.

Software is a basic form of intellectual property but *how* it should be legally protected has been subject to extensive debate. Copyright laws, patents, and trade secrets can all be used to protect software programs, but the efficacy of these forms of protection has been heavily disputed. Patents cover unique processes and functions, but since virtually all software is derivative, patent protection seems inappropriate for software programs. Copyright protection may be more suitable since it does distinguish between ideas and their expression. However, the extent and scope of that protection is unclear, and when a competitor has stepped over the line and infringed on another’s property rights can sometimes be difficult to determine.

From a moral point of view a compelling case can be made for a *natural* right to property. Philosophers often invoke Locke who claimed that labor and effort engendered at least some sort of property rights. Those who work, who invest their time and energy in a project, have a right to the fruits of their labor. Thus, if software companies invest heavy resources and labor in creating a product they should have a limited moral right to the returns which will be generated by this investment by controlling the use of that product. To a certain extent, American copyright and patent laws are grounded in this Lockean conception of property, since they confer long, heavily protected monopolies for product innovations that preclude competitors from cloning or otherwise imitating the protected product.

What we have been describing is the traditional Western conception of *private* property rights. It should be remarked, however, that other countries and traditions adopt a very different approach to property. In general, they put more emphasis on common property whereby some sort of access is granted to all. This is especially true with intellectual property which is conceived by many developing countries as common property. According to Steidlmeier, “developing countries ar-

gue that individual claims on intellectual property are subordinated to more fundamental claims of social well-being . . . [and] that while people may have a right to the fruit of their labor, they have a duty to reward society which practically made the very fruitfulness of labor possible."⁴ This divergence of opinion on intellectual property between American businesses and some of their competitors throughout the world has sometimes been the source of intractable moral problems.

Another source of moral conundrums is the difficulty of differentiating between the property rights of employers and their employees. For example, what portion of an information asset or a software program belongs exclusively to one's present employer and which portion constitutes part of the employee's general knowledge and expertise that can be utilized without impunity in another organization? Does the employee's creative work on a project engender any sort of property rights or are these always surrendered to the employer?

This chapter begins with a relatively straightforward case called, *Piracy on the Internet*. It describes a well-publicized incident in which software products were disseminated on the Internet without permission. There may be agreement that a serious moral transgression has occurred but there is less consensus on the locus of responsibility and a fitting penalty for this misdeed in cyberspace. This case also provides a vehicle for discussing the unconventional viewpoint that software should not be considered as anyone's property.

The next case, "*It's Never Right to Copy Software*," raises similar issues. A young teacher in a poor school district must decide about making unauthorized copies of a software program; unless he does so, the school's new computers will be virtually useless. This case asks the question implied by its title: Is the moral injunction against making illegal copies of software programs absolute or does it allow for exceptions in extenuating circumstances?

In the next two cases the question of ownership is perhaps less clear. In *Whose Program Is This?* an employee takes portions of a successful accounts payable program to her new position. Does her labor on this effort confer any right to appropriate some of this program for her own use, especially when there are no adverse consequences for her employer? And in the case entitled, *Doric Conversion Technologies, Inc.*, a software program is no longer marketed by a company, but is it acceptable for its developer to use this discarded program for his own personal gain?

The next case study, *Software Compatibility and Reverse Engineering*, focuses on two major ongoing legal controversies in the software industry which have significant ethical implications. Both disputes deal with the theme of compatibility. The user community is looking for stan-

dards in software, yet standardization sometimes conflicts with the notion of intellectual property protection which may permit a particular company to have exclusive rights to a technology that is emerging as an industry de facto standard. Can we have common standards and some degree of compatibility without pulverizing the laws that are designed to protect intellectual property? The debate about standards comes to the forefront in the legal dispute of *Lotus v. Borland*, where the primary issue is how much similarity should be allowed between user interfaces. The second conflict which is discussed in this case is the one which took place between Sega Ltd. and Accolade, Inc. The central question here is the following: What can a vendor do to obtain information needed to write software programs that will work on another vendor's hardware platform. In other words, to what extent should reverse engineering be permitted?

The chapter concludes with a provocative case study that adds another layer of complexity to these problems. In the Harvard Business School case, *The IBM-Fujitsu Dispute*, we are forced to consider how the American notion of intellectual property which sets up a temporary monopoly as an incentive to innovation clashes with the more socialistic Japanese view of property that emphasizes its communal nature. This notion becomes the basis for Japan's stress on the diffusion of knowledge in order to stimulate incremental innovations. As a result, two giant companies are caught in the middle of conflicting paradigms of intellectual property protection.

Case 6.1 Piracy on the Internet

In March 1994, the U.S. Attorney in Boston announced that charges would be filed against a Massachusetts Institute of Technology (MIT) student, David LaMacchia, for computer fraud. It was alleged that this 20-year-old student operated a computer bulletin board on the Internet which distributed copies of various copyrighted software programs. LaMacchia was not accused of actually uploading or downloading any of the programs. Also, the indictment made it clear that he did not collect any money or materially profit from this activity in any way.

The problem of "piracy" and pilfered software programs has become quite serious and expensive. According to the Software Publishers Association, worldwide losses from stolen software amounted to a staggering \$7.45 billion in 1994. Moreover, downloading from bulletin boards is now considered to be one of the most common and costly forms of software piracy. Shortly after LaMacchia's indictment Daniel A. Goldman, a student at Brown University, was arrested for alleged