Judicial Understanding of Information Technology: The Case of the Wombat ROMs

R. A. CLARKE

Australian National University, Department of Commerce, GPO Box 4, Canberra, ACT 2601, Australia.

In May 1986, the High Court of Australia handed down its judgment in an important software copyright case. This paper analyses the judgment, and assesses the understanding of computing shown by the nine judges who were involved in the original case and the two appeals.

The conclusion is reached that the information technology industry will encounter significant difficulties and uncertainties as further cases involving technical issues come before the courts.

Received September 1986

1. INTRODUCTION

In 1983, as part of a concerted world-wide effort to insure its technology investment, Apple sued the Australian distributor of a Taiwanese copy of the Apple II. Remarkably, it was the first occasion that an Australian court had been asked to consider whether computer programs were subject to copyright.[9] Apple lost. The result discomfited the industry, and after months of (at times, near-hysterical) pressure, the government amended the Copyright Act to ensure that it extended to computer software. In 1984, on appeal, Apple was successful, although the decision was not unanimous. The distributor exercised his right to take the matter to the final arbiter, the High Court of Australia. In 1986, in another split decision, Apple lost again; the trade press quickly labelled it the 'apple turnover' decision. With the final score standing at 5 judges to 4, there are some important lessons for the information technology industry.

2. OUTLINE OF THE CASE

Apple Computer Inc. manufactures the well-known brand of personal computer called, generically, the Apple II. The family of products was introduced in 1978, and has a significant share of its particular market segment world-wide. By mid-1983 more than 20,000 had been sold in Australia. Some companies, particularly in countries which are not signatories to either of the two International Copyright Conventions, have copied Apple's design, and offered closely compatible alternatives at a lower price. Those which are most closely compatible have, of necessity, either directly reproduced some of Apple's code, or at least achieved the equivalent result using 'reverse-engineering'. Such a machine, manufactured in Taiwan, was imported into Australia by Computer Edge, previously an Apple dealer. It was offered for sale under the name of Wombat, together with Apple II manuals. Apple sued Computer Edge for breach of its copyright in two programs, Autostart and Applesoft, both of which were stored in ROMs in the Apple, and reproduced in a slightly different configuration of ROMs in the Wombat.

At first instance, in the Federal Court on 7 December 1983, the trial judge found in favour of Wombat's distributor, on the grounds that there was no copyright in either source or object code. On appeal, on 29 May 1984, the full bench of the Federal Court, by a 2–1 majority, reversed the decision. The final appeal, to the High Court, decided on 6 May 1986, restored the original judgment, although only by a 3–2 majority.

3. THE RELEVANT LAW

The relevant statute is the Australian Copyright Act 1968, as amended. This is supplemented by a wide variety of case law, particularly of the last 100 years, and arising out of decisions of British courts, of senior courts in some other countries, and of course in the Australian Federal and High Courts.

Secondary sources summarising Australian copyright law include Sterling and Hart[16] and Ricketson.[15] Recent discussions regarding copyright in software in Australia include Lahore,[11] Liberman,[13] Cohen,[3] Clarke,[2] Griffith,[7] Greenleaf[6] and Hughes.[9] The issue has been active in other parts of the world in recent years, notably in the United States – see for example Lechter[12] and Graham.[5]

4. A BASIC MODEL OF THE ISSUE

This article does not attempt an authoritative discussion of the legal aspects of the case, but rather assesses the nature of the judicial understanding and reasoning in relation to some of the fundamentals of information technology.

An adequate model of the key technical issues is provided by Exhibit 1. For the Wombat ROMs to involve an infringement of copyright, the most likely line of argument was that the source program was an 'original literary work' under the Copyright Act, that assembling it into object code and expressing that code in ROMs represented an 'adaptation' in terms of the Act, and that copying the ROMs represented a 'reproduction' in terms of the Act. One other possible line of argument is also shown.

The legal issues were more complicated. The two most accessible expositions are in the judgments of Gibbs (pp. 5–6) and Mason and Wilson (p. 18). Apple's argument in respect of each of the two programs is shown in Exhibit 2. Exhibit 3 represents the issues in the form of a decision table.

The position under United Kingdom law of the same date is unclear. There was a general feeling that computer programs were capable of interpretation as literary

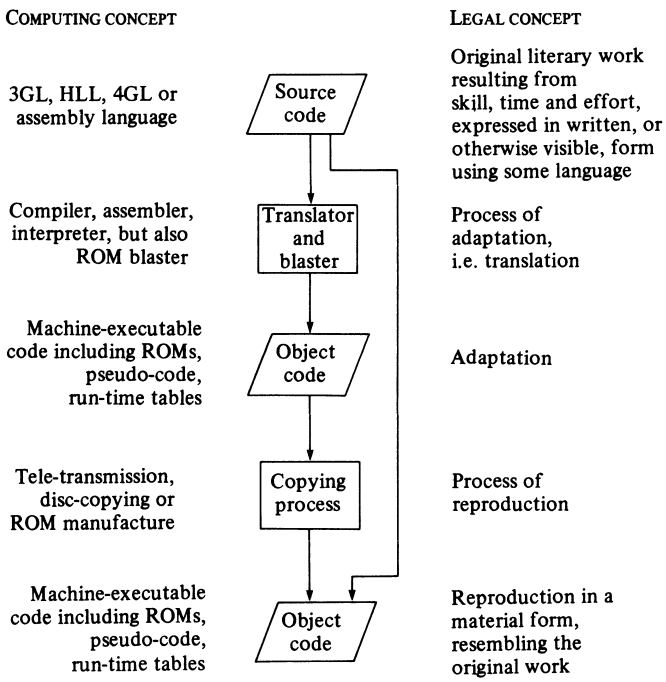

Exhibit 1: A basic model of the process

works, and therefore subject to copyright (e.g. Whitford Committee, chapter 9;[18] Niblett, p. 46;[14] Tapper, p. 18),[17] but the matter had not been tested in the courts. The Copyright (Computer Software) Amendment Act (1985) sought to ensure that programs would be covered; those provisions also await confirmation by the courts.

In the United States, where Apple had commenced its campaign to protect its software, the position was already quite clear. The U.S. Court of Appeal had concluded that 'the copyrightability of computer programs is firmly established after the 1980 amendment to the Copyright Act 1976' (Williams Electronics v. Artic International, 1982). Moreover, 'a computer program, whether in object code or source code, is a 'literary work'... [and]...a computer program in object code embedded in a ROM chip is an appropriate subject of copyright' (Apple v. Franklin, 1983, pp. 17, 18).

[(1) (a) the source program was an 'original literary work'; and
(b) the object program in the Apple II ROMs was a 'translation' and therefore an 'adaptation' of the relevant source program
hence: the object program in the Apple II ROMs was the subject of copyright;
or (2) the object program in Apple II ROMs was itself an 'original literary work' and hence the subject of copyright;]
and (3) the object program in the Wombat ROMs was a 'reproduction' of the object program in the Apple II ROMs;
hence: there was an infringement of copyright.
Alternatively:
(4) (a) the source program was an 'original literary work'; and
(b) the object program in the Wombat ROMs was:
(i) an 'adaptation' of the Apple source program; or
(ii) a 'reproduction' of the Apple source program;
hence: there was an infringement of copyright.

Exhibit 2: Apple's argument.

CONDITIONS				
A: the source program was an 'original literary work'	Y	Y	Y	—
B: the object program in Apple II ROMs was itself an 'original literary work'	—	—	—	Y
C: the object program in the Apple II ROMs was a 'translation' or 'adaptation' of the source program	Y	—	—	—
D: the object program in the Wombat ROMs was a 'reproduction' of the object program in the Apple II ROMs	Y	—	—	Y
E: the object program in the Wombat ROMs was an 'adaptation' of the Apple source program	—	Y	—	—
F: the object program in the Wombat ROMs was a 'reproduction' of the Apple source program	—	—	Y	—
CONCLUSIONS				
Z: there was an infringement of copyright.	Y	Y	Y	Y

Exhibit 3: Issues for resolution.

5. THE JUDGMENTS

5.1 The Trial Judge

Justice Beaumont considered that 'none of the [source or object] programmes are literary works within the meaning of the statute...a literary work for this purpose is something which was intended to afford "either information or instruction or pleasure in the form of literary enjoyment"...the function of a computer program is to control the sequence of operations carried out by a computer' (pp. 22–3). This definition of literary work is from Hollinrake v. Truswell (1894), a case of little significance until its use in Exxon Corporation v. Exxon Insurance Consultants International Ltd (1981).

He also noted 'the omission by the Parliament to make any reference to computers or computing equipment when it determined to extend the scope of copyright protection [to video and sound recordings in 1969]' (p. 23). **In terms of Exhibit 3, he held Conditions A and B in the negative, and hence did not need to consider Conditions C, E or F.** He did find that the Wombat ROMs were copied from the Apple ROMs, i.e. **he resolved Condition D in the affirmative.** Indeed, during the appeal process, Computer Edge did not contest that ruling.

5.2 The Federal Court majority

Several months later, the Full Court, comprising three judges, unanimously concluded that the source programs were 'original literary works'. The Hollinrake authority was regarded as being not sufficiently comprehensive: 'The definition of "literary work" is not exhaustive. It may well take account of modern means of communication and of recording information, which have moved so much (and so rapidly) into the electronic field...There is no necessity for a literary work to be of any literary quality' (Fox, pp. 8, 19). To be a literary work, '[i]t is sufficient if the work supplies information capable of conveying an intelligible meaning and if mental effort and industry is expended in its preparation..."Originality" means that the author's own skill and labour must be involved, though the degree of such skill and labour required is slight' (Lockhart, pp. 29–30).

As far as the question of the language in which the source-code was written is concerned, '[i]t is irrelevant that 6502 Assembly Code is a computer language comprised of three letter mnemonics... It is a highly developed language plainly intelligible to people familiar with it or skilled in its use' (Lockhart, p. 31). **Condition A was therefore resolved in the affirmative.**

Justices Fox and Lockhart further held that the object programs were 'adaptations', 'because they can fairly be described as translations. Transliteration may more precisely explain what happens, but this is plainly comprehended within "translation"' (Fox, p. 21). 'Object code is essentially a mechanical translation of the source programme into another language... Given the source programme the object code version is predetermined by it... When in written form... programmes in object code... are humanly intelligible... [T]hey answer the description of translations of the source code from which they are derived' (Lockhart, p. 37).

It appears that the Justices saw no difference between object programs which were stored on disk or tape, or blasted into ROM, even though, in practice, that generally involves two successive steps: '... the object codes in the Apple II ROMs are adaptations... of the original literary works constituted by the programmes in source codes' (Fox, p. 21). It was also asserted that 'an adaptation of a literary work does *not* itself have to be a literary work' (Fox, p. 22, my *italics*). Nor does it matter whether the code is visible, with or without special apparatus; what matters is that it is recorded in a material form (Fox, pp. 24–5). The versions in Apple ROM were therefore held to be adaptations of the source-code. **Condition C was therefore resolved in the affirmative.**

Lockhart had 'considerable reservations' as to whether the object programs were 'original literary works', but did not need to decide the question (p. 38). Similarly, Fox had no need to consider the matter (p. 22). **Condition B therefore was left unresolved, but in some doubt.**

Lockhart found that the Wombat ROMs are 'reproductions in a material form' of the Apple ROMs (p. 40). Fox concluded similarly that **Condition D was fulfilled** (pp. 23–5). Hence the chain of original literary work, adaptation and reproduction was established (Conditions A, C and D in Exhibit 3), and an infringement of copyright had occurred (Conclusion Z).

Conditions E and F were left unresolved, since the matter had already been decided in Apple's favour.

5.3 The Federal Court minority

Justice Sheppard agreed that 'the programmes as originally written in source code were literary works' (p. 11), and discounted the Hollinrake judgment. **Condition A was therefore supported.** He argued that '... the programmes in object code are not literary works... [because] it is only the machine itself... which can "understand" or "see", and thus deal with, the object code' (p. 14), i.e. they fail the 'visible form' criterion. **Condition B was therefore denied.**

Sheppard argued further that 'adaptations of literary works, like literary works themselves, must... be capable of being seen or heard' (pp. 17–18). This is far more restrictive than the 'human intelligibility' test applied by the majority judges. Since they fail this test, Sheppard concluded that the object programs are not adaptations, **denying Conditions C and E.** None of the four Federal Court judges offered an opinion as to whether Apple's object programs were reproductions of its source programs, but Sheppard pointed out that that would have been insufficient to support Apple's case, since the reproduction of a reproduction does not infringe copyright (p. 13).

It does not appear that Justice Sheppard addressed Condition F (whether the object program in the Wombat ROMs might be a reproduction of the Apple source program), although Exhibit 3 suggests that he needed to do so. It appears unlikely that he would have affirmed it, since it appears to fail the test that a reproduction 'resemble' the original literary work from which it was copied, and an object program in ROM bears little resemblance to a source program in any form. He concluded that the Wombat ROMs copied a mere reproduction of an object program in which no copyright existed; hence no infringement took place. Sheppard was almost apologetic about his conclusions e.g. 'Notwithstanding that my sympathies are with the appellants [Apple]... '(p. 18). Moreover, he felt that 'the position may be entirely different in relation to application programmes... [a]nd it may vary depending on the circumstances of each case' (p. 20). Given his 'visible form' test, it is unclear how that could be so.

5.4 The High Court minority

Two years later, Justices Mason and Wilson also dismissed the Hollinrake argument, partly on the grounds that 'the appellants would be hard-pressed to deny that the source programs... were intended to afford information and instruction' (p. 19). They expressly concluded that the language into which a source-code language is translated did not need to be a 'human language' (or, as a computerist might say, a 'natural language'), and hence that 'translation' was to be interpreted in a broad sense (p. 22). They comment that '[t]he question whether an adaptation of a literary work must itself be a literary work is more difficult of resolution' (p. 22). To the layman it seems remarkable that such an apparently fundamental definitional question remains open after decades of case law and amendment legislation. Unlike Justice Fox, they concluded that it *is* a requirement of an adaptation that it be itself a literary work. However, the judges interpreted the qualifying condition as a literary work to be 'material' form, rather than the more constrictive 'written' form (p. 23). The reasoning was therefore different, but the result the same. They concluded that 'we are broadly in agreement with the reasons and conclusions of Fox J. and Lockhart J.' (p. 17), i.e. 'the Wombat computers contain a reproduction in a material form of an adaptation of an original literary work' (p. 24). **Conditions A, C and D were therefore resolved in the affirmative, but B, E and F were left unresolved.**

5.5 The Chief Justice

Chief Justice Gibbs considered that the source programs were in 'visible form' and, in satisfaction of the Hollinrake test, 'afford instruction to the operator keying in the machine that will convert the source code to object

code' (p. 7). Unfortunately, it is almost impossible to extract any meaning from those words. The biggest problem is the meaning of 'keying in the machine'. With two interpolations, 'keying the source code into the machine' could produce a meaningful interpretation. However, the source code's primary function is to convey information to the translating software (in this case the 6502 Assembler), rather than to 'afford instruction' to any human operator. Whether or not I can understand the reason, the Chief Justice concluded that the source programs were 'literary works'. They also satisfied the test of originality (pp. 7–8), and hence **Condition A is fulfilled.**

However, Justice Gibbs held that '...the object programs embodied in the ROMs...were not literary works', and hence **Condition B is negated** (p. 8). Rather than applying the Mason/Wilson test of 'material form', he applied the narrower condition of 'expression in print or writing': 'It seems to me to be a complete distortion of meaning to describe electrical impulses in a silicon chip, which cannot be perceived by the senses and are not intended to convey any message to a human being and which do not represent words, letters, figures or symbols as a literary work; still less can a pattern of circuits be so described' (p. 9), and again 'I have not found anything [in the cited authorities] that has pursuaded me that a sequence of electrical impulses in a silicon chip, not capable itself of communicating anything directly to a human recipient, and designed only to operate a computer, is itself a literary work' (p. 13). This seems to be somewhere between the Sheppard 'visible form' test and the Lockhart 'human intelligibility' test.

In assessing whether an object program is an 'adaptation' of the source code, Gibbs argued for a narrow definition of the word 'translation'. Moreover, he concluded that 'an adaptation must itself be a "'work"' (pp. 10–11), which is the opposite of Justice Fox's conclusion, but consistent with his brother judges Mason and Wilson. **Condition C is therefore denied.** Moreover, this line of argument further implies that 'the Wombat object programs [in ROM] themselves were not adaptations of the [Apple] source programs' (p. 11), which represents **denial of Condition E.**

On the question as to whether the Wombat ROMs are 'reproductions' of the Apple source programs, Gibbs found insufficient resemblance between the Apple originals and the Wombat product (pp. 11–12). **Condition F was therefore explicitly denied.**

From Exhibit 3, it is clear that **Condition D is now irrelevant** since both Conditions B and C were resolved in the negative. However, 'there is no doubt that the programs which appear in the Wombat ROMs...are the same as those which...are used in the relevant Apple computer' (p. 3).

According to the Chief Justice, the Wombat ROMs did not infringe any copyright.

5.6 Justice Brennan

Condition A was held in the affirmative. 'The source programs were the product of substantial originality and skill, they were prepared as instructions for the manufacture of Apple II ROMs, they were in writing and they conveyed meaning at least to computer scientists and technicians. That is sufficient to bring them within the scope of literary copyright' (p. 30).

In assessing whether the resulting object code was a literary work, Brennan applied a narrow definition of materiality akin to Sheppard's: 'A material form is a form which can be perceived by the senses...The electrical charges which constitute the object programmes cannot be seen or touched or heard, or, if they can, then they do not communicate the letters of the original literary work, the source programmes...The object programmes are not literary works' (p. 32). **Condition B is therefore denied.**

As to whether the object code is an adaptation or translation, Justice Brennan argued that '[i]t strains the meaning of "language" to include within its denotation the alphabetic symbols in which the source programmes were written...It is difficult to divorce "language" from human speech, and a means of communicating ideas which does not consist of words is not properly to be described as a language..."6502 Assembly Language" is not a language; it is a code...the electrical charges which constitute the object programmes are clearly not a language...The machine has no comprehension of thought which it is the essential purpose of language to convey' (p. 34).

In addition, Brennan joined his brother judges in opposing the Fox assertion, and recognising 'the necessity of the "adaptation" being itself a "work" (p. 35, based on s. 31(1)(a)(vii)). Since the object code is neither a translation nor a literary work, **Conditions C and E are emphatically negated.**

'There is no doubt that the circuitry of the Wombat silicon chips was derived from the circuitry of the Apple II ROMs' (p. 26). Hence, if relevant, **it is likely that Condition D would be accepted.** Finally, 'the notion of reproduction...connotes a resemblance between the work in which copyright subsists and the work which is copied from it...there is no resemblance between the Wombat ROMs...and the written compilations of the...source programs' (pp. 37, 40). Accordingly, **Condition F is negated.**

5.7 Justice Deane

Justice Deane pursued a different line of argument entirely. He commenced with a premise of novelty to a computerist: that 'the written program in source code is directed to a human reader and not to a machine. Its essential function is to record and communicate programming instructions' (p. 41). The second sentence is true, but incomplete: a program's essential function is to communicate to a translating program in a machine; only secondarily is it for communication to a human (and, not infrequently, only to the human who wrote it).

The following passage confirms a potentially crucial misunderstanding of the nature of program language translation: 'the written expression...in source code consisted essentially of *instructions...to be read and followed by a human reader in his or her operation of an assembler* to generate the series of electrical impulses or charges constituting the relevant operational programme' (p. 41, my italics). The human does not read or follow the instructions; he or she merely feeds them to the assembler (translation program), which generates the machine-executable instructions (electrical impulses or charges) corresponding to the assembly language instructions.

	Beaumont	Fox and Lockhart	Sheppard	Mason and Wilson	Gibbs	Brennan	Deane
A	No; Hollinrake criterion	Yes; Hollinrake irrelevant	Yes; Hollinrake irrelevant	Yes; Hollinrake irrelevant	Yes; Hollinrake satisfied	Yes	—
B	No; Hollinrake criterion	In doubt	No; 'visible form' criterion	—	No; 'print or writing' criterion	No; 'human perception' criterion	No; 'visible form' criterion
C	—	Yes; 'human intelligibility' criterion; not 'literary work' criterion	No; 'human perception' criterion; 'literary work' criterion	Yes; broad 'translation' definition; 'literary work' criterion; 'material form' criterion	No; narrow 'translation' definition; 'literary work' criterion	No; narrow 'translation' definition	No; 'directions to a human'
D	Yes	Yes	Yes?	Yes	Yes?	Yes?	Yes
E	—	—	No; 'human perception' criterion	—	No; narrow 'translation' definition	No? narrow 'translation' definition; 'human perception' criterion	No; 'directions to a human'
F	—	—	—	—	No; 'resemblance' criterion	No; 'resemblance' criterion	No; 'directions to a human'
Z	No	Yes (ACD)	No	Yes (ACD)	No	No	No

Exhibit 4. Resolution of the issues

Based on what appears to me to be an erroneous assumption, the Justice then argued that the program was merely a set of instructions, like instructions for constructing a Meccano model, or a recipe in Mrs Beeton's cookery book (Cuisenaire v. Reed, 1963). The act of one human following the directions of another has long been held not to infringe copyright.

The following passage appears to support Condition A: 'To reproduce in a material form or adapt the actual written instructions or directions constituting a source programme would, plainly enough, involve infringement of the copyright in the written programme.' However, later on the same page he says: 'I find it unnecessary to determine whether a written expression of the...programmes in source or in object code constituted...literary works' (p. 46). **Condition A was accordingly unresolved.**

Condition B was decided in the negative: 'I am unable to accept the proposition that...the actual series of electrical charges constituting each operational programme was itself an original literary work when "embodied" in a ROM or ROMs...because the re-arrangement of electrons in a programmed ROM is not visible to the human eye' (p. 47). The acceptance of all Conditions C–F would involve recognition of object programs in ROM as either adaptations or reproductions of a written expression of a programme. He explicitly denies this: '[the programmed Apple II or Wombat ROM] resulted from the following of the directions of [the written expression of the programme in source code]; it was not, however, a reproduction or adaptation of the literary work embodying those directions' (p. 45). **All of Conditions C, E and F are therefore negated.** The argument appears to be undermined by the false premise discussed earlier. However, the tenor of the argument is such that, even if the premise had been corrected, it seems unlikely that Apple could have negotiated the minefield and proved an infringement.

6. SUMMARY OF THE JUDGMENTS

The judges were able to agree on few specifics arising from the case. Not only did they reach different conclusions, but the reasoning differed markedly. For a reconciliation against the Decision Table in Exhibit 3, see Exhibit 4.

7. JUDICIAL UNDERSTANDING OF THE TECHNICAL ISSUES

Matters relating to information technology have seldom reached the High Court, and, reasonably enough, the judges had some difficulty in understanding the technicalities sufficiently to abstract the key features. One of the functions of counsel is to provide the court with relevant information, both technical and legal. The inherent conflict of interest between the barrister's duties to educate the court, and to represent his client's interests, is conventionally ignored. There is a small pool of barristers in Australia with appropriate experience, and a number with dual qualifications in computer science and law. Both parties availed themselves of the services of one or more such counsel.

The judgments reveal that many aspects of computing were appreciated by at least some of the judges. The judgment of Judge Lockhart (in the majority on the full bench of the Federal Court) contains a lengthy, very clear and, with a number of small qualifications, correct exposition of the nature of the relevant technology (pp. 5–18). This even includes thoroughly usable explanations of RAM, ROM, EPROM, firmware and disassembly.

Judge Sheppard's judgment contains a shorter, less complete, but similarly only lightly flawed description (pp. 3–9).

Exhibit 5 is a collage of definitions and descriptions which appear in the other judgments, and which, in this author's assessment, are fair, reasonable and, in several cases, erudite.

* 'A computer program is a set of instructions designed to cause a computer to perform a particular function or to produce a particular result' (Gibbs, p. 1).
* 'A program is usually developed in a number of stages. First, the sequence of operations which the computer will be required to perform is commonly written out in ordinary language, with the help, if necessary, of mathematical formulae and of a flow chart and diagram representing the procedure.

'Next there is prepared what is called a source program. The instructions are now prepared in a computer language... The source code...cannot be used directly in the computer, and must be converted into an object code, which is "machine readable", i.e. which can be directly used in the computer. The conversion is effected by a computer, itself properly programmed.

'The program in object code, the object program, in the first instance consists of a sequence of magnetic impulses which are often first stored on a magnetic disk or tape, and which may be stored permanently in a ROM ('read only memory'), a silicon chip which contains thousands of connected electrical circuits. The object code is embodied in the ROM in such a way that when the ROM is installed in the computer and electrical power is applied, there is generated the sequence of electrical impulses which cause the computer to take the action which the program is designed to achieve.

'The pattern of the circuits in the ROM...cannot be seen with the naked eye...However the sequence of electrical impulses may be described either in binary notation...or in hexadecimal notation...and it is possible to display the description on the visual display unit of the computer, and to print it out on paper' (Gibbs, pp. 1–3).
* 'Converting the program into the...object program...was achieved by using a computer to reduce the 6502 assembly code in which the...source program was expressed into machine language consisting of a series of electrical impulses which were then able to be stored on a magnetic disc or tape or permanently installed in the...computer on ROMs' (Mason and Wilson, p. 17).
* 'Binary notation and hexadecimal notation are conventional ways of representing in writing an object program, but the object program is not the writing. The object program is the sequence of electrical charges and its existence and use are not dependent on a written representation' (Brennan, p. 28).
* 'The process of recording the object program involves the application of the sequence of electrical charges to the chip, so that the minute fusible connectors in the chip are burned out or charges are applied to minute insulated capacitors in the chip' (Brennan, p. 29)
* 'The actual series of electrical impulses or charges constituting the operational object programme may be in a transient state or "stored" in any one of a number of ways including temporarily in the RAM of an activated computer and semi-permanently or permanently on a magnetic tape or disc or in a ROM.

'Once the first set of charged or programmed ROMs had been produced, their functional qualities could be directly reproduced or copied by mechanical means' (Deane, p. 42).

Exhibit 5. Convincing definitions in the High Court Judgment

However, a variety of errors and misapprehensions appear in the judgments. Exhibit 6 shows a number of examples from the High Court judgments. At the outset, counsel confused the court by introducing the terms 'high-level' and 'low-level language', which were irrelevant to the case. They then insisted on talking throughout about 'object code'. 'Object code' is the result of translation (by compilation or assembly) from 'source code'. It is entirely feasible to write a program directly in 'machine code', in which case there is no translator, and no object code. Hence it is more conventional, and sensible, to talk of the active code as 'machine', 'binary', or 'executable' code. The only stage at which the term 'object code' was pertinent to the case was in establishing whether the executable code was a translation from copyright source code (i.e. in Issue B).

The next blunder was in an 'agreed statement of facts', which the parties to the action provided to the court. In it they defined the elements of an assembly code as:

(*a*) 'labels identifying particular *parts* of the program' (Gibbs, p. 2, my italics). In an assembly language, labels identify not parts but points (although the statement could be true in respect of block-structured languages like Pascal);

(*b*) 'mnemonics each consisting of three letters of the alphabet and [each] corresponding to a particular operation *expressed in 6502 Assembly Code* (*the code used*)' (Gibbs, p. 2, my italics). This segment would have been meaningful if the italicised words had been replaced with 'supported by the 6502 instruction-set';

(*c*) 'mnemonics identifying the register in the microprocessor *and/or the number of the instruction in the program to which the operation referred to in* (*b*) *related*' (Gibbs, p. 2, my italics). Reasonably enough, the Justices were entirely unclear as to what the italicised clause meant.

After that inauspicious start, the High Court had little difficulty in creating further confusion for itself. For example, one of the judgments asserted that the recording of the program on blank silicon chips was practically irreversible. Both ROMs and EPROMs were used in the Wombat (and the judgments continually refer to both, even though the two appear to be equivalent for the purposes of the case). The essential distinction between the two is, of course, that with the latter the recording *is* reversible.

Several of the Justices were quite concerned about whether the original source-code could be recovered from the executable code in the ROMs, but they failed to understand what such a process involves. It is possible to create from executable code a semantically equivalent (and isomorphic or similarly structured) assembly-language program. It is *not* possible to recover the labels which the original source code used, nor the comments (or 'instructions', as one judgment inconveniently called them). Nor is it possible to distinguish between assembly statements which were hand-coded and those which were generated by a macro. Technically, none of that really matters, but legally perhaps it does. And how would the court have coped with the far more difficult concept of de-compilation?

Two of the Justices said that it was unclear as to whether the object program was the series of electrical impulses stored in the ROMs, or the written description in binary or hexadecimal notation. Conventionally, executable code is a set of instructions which, when successively loaded into a machine's instruction register,

* 'Next there is prepared what is called a source program. The instructions are now expressed in a computer language – either in a source code (which is not far removed from ordinary language, and is hence called a high level language) or in *an assembly code* (*a low level language, which is further removed from ordinary language than a source code*), or successively in both' (Gibbs, p. 1, my italics).

  Source code may be in a high-level or an assembly language; the two dimensions (high-level/low-level and source/object) are disjunct. Moreover, a source-code may be (and sometimes is) expressed in machine-executable form, in which case there is arguably no object code (since there is no translating program for it to be the 'object' of), or alternatively the source and object are the same.
* Element of an assembly code (*a*): 'labels identifying particular *parts* of the program' (Gibbs, p. 2, my italics). In an assembly language, labels identify points rather than parts (although the statement could be true in respect of some higher-level languages like Pascal).

  This and the next two inaccuracies derive from the parties' 'statement of agreed facts' provided to the original trial judge (Beaumont, pp. 14–15).
* Element of an assembly code (*b*): 'mnemonics each consisting of three letters of the alphabet and [each] corresponding to a particular operation *expressed in 6502 Assembly Code* (*the code used*)' (Gibbs, p. 2, my italics). This segment would be meaningful if the italicised words were replaced with 'supported by the 6502 instruction-set'.
* Element of an assembly code (*c*): 'mnemonics identifying the register in the microprocessor *and/or the number of instructions in the program to which the operation referred to in* (*b*) *related*' (Gibbs, p. 2, my italics).

  In the original judgment (Beaumont, p. 15), the words 'the instruction' were used where Gibbs uses the word 'instructions'. But even in the original, it is unclear what the sentence means.
* 'When [a source program] is *compiled*, it is keyed into a computer...The relationship between the source program and the object program is...known to and used by the *compiler* of the source program who so *compiles* the source program that the desired object program will be produced when the source program is keyed in' (Brennan, p. 27, my italics). Judge Brennan, apparently unaware of the technical meaning of the term, consistently and confusingly used the term 'compile' where a word such as 'prepare' would have sufficed. (The term 'compilation' also has a technical meaning in copyright law, being one kind of literary work, e.g. a railway timetable. This meaning seems of little relevance in this case.)
* 'The recording of the program [upon blank silicon chips] is practically irreversible' (Brennan, p. 29). Both ROMs and EPROMs were used in the Wombat, and all judgments continually refer to both, rather than treating ROM as a generic term. The distinction between the two is, of course, that the latter are erasable and re-programmable.
* 'A printout of the text of [an assembly language] source program (other than instructions which are not keyed in) can also be recovered [from an object program contained in ROM]' (Brennan, p. 29); and 'whilst it is possible to extract from the computer a reconstruction of the mnemonics that were employed it is not possible to reproduce the labels and instructions' (Mason and Wilson, p. 17). Although a dis-assembler may recover a semantically equivalent (and isomorphic) version of the original source-program, it can recover neither arbitrary labels, nor comments ('instructions', in Mason and Wilson's inconvenient terms). Nor can it distinguish between hand-coded assembly statements and those generated by a macro.
* 'The written expression...in [assembly language] source code consisted essentially of instructions...to be read and followed by *a human reader in his or her operation of* an assembler to generate the series of electrical impulses or charges constituting the relevant operational programme' (Deane, p. 41, my italics). This is simply wrong: to the person keying the source-code into a computer, the code is not instructions, but merely character-strings separated by line-feeds. Deletion of the italicised words would produce a reasonable (if inelegant) statement, but would completely invert the meaning.
* 'Whether it is the series of electrical impulses stored in the ROMs or the written description in binary or hexadecimal notation that is truly the object program is not entirely clear' (Mason & Wilson, p. 17). Conventionally, an object program is a set of directly executable instructions resulting from translation from a source program, irrespective of the form in which the instructions are stored. (Clearly, the definition of 'directly executable' requires some care, but a consensus formulation should be fairly readily achievable.)
* 'Wozniak was an officer of the first respondent [Apple]' (Mason & Wilson, p. 19). The learned Judges appear to have no sense of history!

Exhibit 6. Some errors and misapprehensions in the High Court judgment

causes the intended function to be performed. The form in which the instructions are stored is irrelevant.

8. IMPACT ON THE JUDGMENTS

It is not the purpose of this article to assess whether the 'apple turnover' case involved a miscarriage of justice. However, it is appropriate to give brief consideration as to what extent these errors and misunderstandings contributed to the decision.

Because of the complexity of copyright law, most of the misunderstandings do not appear to have had any direct effect on the result. For example, Justice Brennan's misapprehension that an assembly-language source program could be recovered from 'object code' in ROM could have been important in determining his judgment on Condition F. However, since he applied the very restrictive test of 'resemblance', and found no resemblance between the ROMs and the written source, the danger remained latent.

However, one error does appear to have had a significant impact. Gibbs considered Condition A fulfilled since the source programs 'afford instruction to the operator' (p. 7). Deane disallowed Conditions C, E and F because 'source code consisted essentially of instructions...to be read and followed by a human reader...[and]...[the programmed ROM] resulted from the following of the directions of [the source code]' (pp. 41, 45).

Written source-code is not a set of directions; not to a computer, and certainly not to a human. Although a suitably trained human can infer from it what the effect of the resulting machine-executable program will be, the written source code is merely a set of symbols punctuated by line-feeds. When read, as inanimate data, by a suitably programmed translator, the source code will result in machine-executable code which *does* have the ability to direct a machine's actions. However, there is no sense in which such a language directs any human's actions. In view of the fact that the same remarkable misunderstanding occurs in two judgments, the inescapable con-

clusion is that counsel were at fault. Denied their premise, Gibbs and Deane may well have reached their conclusions by other routes. None the less, the actual reasoning in their judgments is undermined by the error.

9. THE NEED FOR AUTHORITATIVE DEFINITIONS

Three of the judgments referred to definitions of source and object code arising from the U.S. National Commission on New Technological Uses of Copyright Works. It therefore appears that, if there existed an annotated glossary of technical terms with at least national and preferably international standing, it might be used by the courts. Glossaries have been prepared by the International Standards Organisation in 1982 and the World Intellectual Property Organisation in 1978 and 1984 (although they may not prove to be very helpful.)[10,19] Alternatively, standard reference works could be consulted, such as the *Encyclopaedia of Computer Science and Engineering* (Ralston, 1983) or the *Dictionary of Computing* (Oxford, 1983).

10. HOW DIFFICULT WAS THIS CASE?

The Wombat case involved only assembly code, and machine-executable code in ROM. It even appears that no consideration was given to macros, which have an impact on the relationship between source and object programs, and the nature of the disassembly process. Even if the Australian and U.K. Copyright Amendment Acts have successfully clarified many of the issues which arose in this case, there are myriad additional technical complexities which will arise in future cases. Compilation of non-isomorphic 'third-generation', 'higher-level' languages, and interpretation of both 'third-generation' and 'non-procedural' languages have been commonplace for between 20 and 30 years. More recent innovations have included: multiple-phase arrangements involving a generator, compiler and linker; run-time interpreters for low-level pseudo-codes; run-time table processors for some kinds of application generators; parameterised application software packages; heavy dependence on separately compiled sub-routines, copyright in which may be owned by the same organisation, another or several other organisations, and which may be embodied in removable magnetic media, ROM or a distinct processor integral to the configuration; re-entrant code; virtual memory and swapping; down-line loading of software to workstations; and software packages distributed between workstations, cluster controllers and central servers.

11. IMPLICATIONS FOR FUTURE CASES

The Australian Copyright Amendment Act 1984 came into operation on 15 June 1984.[4] As Justices Mason and Wilson noted, 'similar issues that may arise in future proceedings will have a very different statutory context for their resolution' (p. 15). However, that Act will also require interpretation, and it is likely that some of the High Court's various arguments may be invoked in future cases. Some of the issues likely to arise include the manner in which the source program is originally expressed (in particular whether it is first, or ever, expressed 'in writing', or indeed in any human-readable form); whether an adaptation must itself be a literary work; whether all the different forms of translation (e.g. compilation, interpretation and generation) constitute 'adaptation'; whether all the different forms in which machine-executable code is stored constitute 'reproductions in a material form'; and whether, as Justice Deane contended, programs are (in law, even though not in fact) merely directions.

Moreover, all of this discussion relates only to the question of rights relating to programs. Much data has now been captured into machine-readable form, and it is not entirely clear what rights pertain to such on-line databases. A great deal of authorship is now undertaken with the assistance of computer-based equipment. This has applied for some time to text of all kinds, ranging from industrial and academic research to fiction and poetry. With the enormous improvements in image creation and manipulation, it applies increasingly to graphical creation. Sound, both human voice and music, is undergoing a similar revolution.

These questions relate to contemporary technology and should have been resolved by now, yet they still present the courts with great difficulties of comprehension. Beyond them lie such impending issues as rights relating to computer-originated works, and the boundaries between computer-origination and computer-assisted human authorship.

12. CONCLUSION

That not all judges are opposed to development within the law is shown by several statements at the Federal Court level: 'It may well take account of modern means of communication and of recording information, which have moved so much (and so rapidly) into the electronic field... There has I think long been a tendency (not invariably observed) to apply the language of the statutory law governing copyright in a practical manner, consistently with the needs of time, and the then current concepts' (Fox, pp. 8, 20); 'Courts have generally construed copyright legislation mindful of changes in ideas and advances in technology... [C]opyright legislation should be construed liberally and with a view to the furtherance of justice. In particular, such legislation should be interpreted to keep pace with technological innovation [provided that the subject matter] is conformable with the principles developed by the courts over many years of experience' (Lockhart, pp. 33, 54).

A few months earlier, the High Court had re-affirmed its strong disinclination to make new law: '[Requiring that reasons be given for decisions made adverse to the interests of an individual]... is a change which the courts ought not to make, because it involves a departure from a settled rule on the grounds of policy which should be decided by the legislature and not by the courts' (Public Service Board of N.S.W. v. Osmond, 1986, p. 11).

In the Apple case, Justice Deane concluded that '... general questions of the existence and extent of copyright protections in a new field cannot be determined by reference to notions of what may or may not be fair in the circumstances of a particular case' (p. 48); and the Chief Justice said that 'Important questions of policy arise when it becomes necessary to consider whether copyright protection should be extended to such a thing

as ROM, and the courts can act only within the existing statutory framework' (p. 13). While rushing the Copyright Amendment Act through Parliament in Mid-1984, the then Attorney General committed his Government to 'now proceed as a matter of high priority to a consideration of policy for the long term through an appropriate form of enquiry' (Second Reading Speech, 4 June 1984). Soon after, there was a cabinet reshuffle. No such enquiry was commenced. Despite these explicit statements that a majority of the High Court, including the Chief Justice, prefer stagnation of the law to judicial innovation, no such enquiry is mooted. To Australian politicians, the ambiguities arising from the 'apple turnover' judgment are uninteresting and complicated economic and legal matters, rather than a political issue offering ballot-box payback.

At least in Australia, the information technology industry must expect that the law will continue to be interpreted in its traditionally narrow and pedantic fashion. As evidenced in this case, there are a wide variety of angles which imaginative counsel from both sides can and will pursue. Aided and abetted by expert witnesses and barristers, judges will perpetrate a wide range of partial and even complete misunderstandings. With the efflux of time, pioneer cases will fulfill the function of educating the judges in key aspects of the technology. Moreover, in due course, lawyers of the 'computer-literate' generation will reach the bench. In the interim, apart from the possibility of specific legislation in some areas, one of the few options available is the establishment of agreed nomenclatures, such that the scope for expert witnesses to cancel one another out is constrained, and so indeed is the scope for barristers to pull the wig-wool over judges' eyes.

It would be grossly unwise for any organisation to expect that its presumed, even commonly accepted, moral rights necessarily translate into legal ones. This conclusion may well apply as much to contract, sale of goods, consumer protection and trade practices law as to copyright.

REFERENCES

1. J. W. K. Burnside, 'The legal implications of computers.' (1981), 55 Aust. L.J. 79.
2. R. A. Clarke, 'Arguments for software protection.' *Australian Computer Bulletin*, 24–9, (1984).
3. G. M. Cohen, 'Computer Programs – Does the Law Provide an Adequate Protective Mechanism?' (1982), 56 Aust. L.J. 219.
4. P. Crisp, Address to the W.A. Society for Computers and Law, Attorney-General's Department, Canberra, 22 August 1984.
5. R. L. Graham, 'The legal protection of computer software.' *Comm ACM* **27**(5), 422–6 (1984).
6. G. W. Greenleaf, 'Software copyright: one rotten Apple?' *International Media* L. (1984).
7. P. B. C. Griffith, 'Intellectual property protection of computer programs.' *Proc. ACC '84, Sydney*, Australian Computer Society, November 1984.
8. K. Hughes, *Copyright in Computer Software*. Australian Copyright Council, Bulletin No. 38, November 1981
9. K. Hughes, *Computers and Copyright*. Australian Copyright Council, Bulletin No. 49, August 1984.
10. ISO, *Data Processing – Vocabulary*. Standard Handbook No. 10. International Standards Organisation (1982).
11. J. C. Lahore, 'Computers and the Law: The Protection of Intellectual Property.' 9 Fed. L. Rev. (1978).
12. M. A. Lechter, 'Protecting software and firmware developments.' *IEEE Computer*, 73–82 (1983).
13. A. Liberman, 'The Protection of Computer Software in Australia.' 9 Aust. Bus. L. Rev. 234 (1981)
14. B. Niblett, *Legal Protection of Computer Programs*. Oyez, London (1980).
15. S. Ricketson, *The Law of Intellectual Property*. Butterworths, London (1984).
16. J. A. L. Sterling & G. E. Hart, *Copyright Law in Australia*. Legal Books, (1981)
17. C. Tapper, *Computer Law*. 3rd ed., Longman, London (1983).
18. Whitford, *Copyright and Design Law*. Cmnd 6732 HMSO, London (1977).
19. WIPO, *Model Provisions for Protection of Computer Software*. World Intellectual Property Organisation (Geneva, 1978).

Statutes

Copyright Act 1968 (Commonwealth of Australia)
Copyright Amendment Act 1984 (Commonwealth of Australia)
Copyright Act 1956 (United Kingdom)
Copyright (Computer Software) Amendment Act 1985 (United Kingdom)
Copyright Act 1976, 17 U.S.C. 101 ff (United States of America)

Cases

Apple v. Computer Edge (1984) 53 A.L.R. 225 (29 May 1984) Fox J. 229–34; Lockhart J. 247–53; Sheppard J. 271–4
Exxon Corporation v. Exxon Insurance Consultants International Ltd (1981) 3 All E.R. 241 and (1982) 1 Ch. 119
Federal Court Judgment No. G130 of 1983, 7 Dec, 1983, Apple Computers v. Computer Edge (Beaumont J.)
Federal Court Judgment No. G405 of 1983, 29 May 1984, Apple Computers v. Computer Edge (Fox J., Lockhart J. and Sheppard J.)
High Court of Australia Judgment No. F.C. 86/006 (21 February 1986) Public Service Board of N.S.W. v. Osmond
High Court of Australia Judgment No. F.C. 86/017 (6 May 1986) Computer Edge v. Apple Computer (Gibbs J. 1–14; Mason & Wilson JJ. 15–25; Brennan J. 26–40; Deane J. 41–48)
Cuisenaire v. Reed [1963] V.R. 719
Hollinrake v. Truswell [1894] 3 Ch. 420
U.S. Court of Appeals for the Third Circuit Judgment No. 82–1582 (August 30, 1983) Apple v. Franklin
Williams Electronics Inc v. Artic International Inc [1982] 685 F.2d 875