The New Software Copyright Law

ALAN BUNDY* and HECTOR MACQUEEN†

Department of *Artificial Intelligence and †Private Law, University of Edinburgh, Edinburgh, UK

1. INTRODUCTION

On January 1st 1993 a new UK law slipped onto the statute books, with remarkably little public comment. Since this law could have a significant impact on the market for computer products in the UK, it was especially surprising that there was little debate about it in the computing community.

The new law is called ‘The Copyright (Computer Programs) Regulations 1992’. To understand this law, and the changes to software copyright that it will bring, it is necessary to understand some of the events which led up to its introduction.

2. METHODS OF PROTECTING SOFTWARE PRODUCTS

If you invest a lot of time and effort in the development of a new commercial product then you do not want a rival to copy your product and undercut your price. This possibility acts as a disincentive to the development of new products. To prevent it governments have introduced a variety of intellectual property rights: patents, copyright, design rights and trademarks.

Patents, for instance, protect new inventions. The protection usually lasts 20 years and applies to any product which incorporates the new invention. Patents have to be specially applied for and paid for, and are only granted if, after investigation by the Patent Office, they are found to be novel, inventive and capable of industrial application.

Copyrights, on the other hand, protect literary, dramatic and musical works. The protection lasts until 70 years after the author’s death and is specific to the particular expression (this is a legal technical term) of the work. Copyrights come into existence automatically on the creation of a work and do not have to be especially applied for or paid for.

Until recently copyright was the only form of protection for computer programs. Programs are regarded as a kind of literary work. Treating programs as literary works is a convenient fiction. It means that all existing copyright law automatically applies to them, including the international Berne treaty, without the need for new laws and difficult international negotiations. Unfortunately, it is a fiction. Computer programs differ enough from real literary works that the law requires some special amendments to make it fit their case.

Computing techniques were traditionally regarded as unpatentable. They were seen essentially as mathematical algorithms, which have always been excluded from patent law. However, this situation is gradually changing. The change started with attempts to patent hybrid inventions consisting partly of programs and partly of physical devices. In 1981 the US Supreme Court allowed a hybrid invention to be patented in the case of Diamond v Diehr. Since 1981 the number of software patents in the USA has gradually increased and patents have begun to be issued on purely software inventions. The European Patent Office has started to follow the US lead. In turn the UK Patent Office and courts have now also begun to consider when programs are patentable. This change has proved highly controversial, with organizations like the League for Programming Freedom campaigning to have it reversed.

In this article we will be concentrating on the situation with software copyright and saving the patent controversy for another day.

3. ‘LOOK AND FEEL’ DISPUTES

Software copyright law has also been an arena for controversy. The main battles have been a series of so called ‘look and feel’ disputes in the USA. The legal argument has been whether the look and feel of a computer program can be protected by copyright. For instance, Microsoft produced a windows system providing a user interface similar to the Apple Macintosh. Apple sued Microsoft for breach of copyright. No-one disputed that Microsoft’s system used completely different code from Apple’s. So there was no copyright infringement of a particular piece of program. Apple’s claim was that the overall appearance of their system had copyright and that this had been infringed. In the latest round of a long fight Apple’s claim has been dismissed by the judge; however, they plan to appeal.

In a similar case, Lotus Development Corp sued Borland for producing a spreadsheet which can be customised to have a similar interface to the Lotus 1–2–3 spreadsheet. This case has gone the other way; Lotus have won and Borland have had to withdraw the 1–2–3 look-alike feature from their spreadsheet. Again, however, Borland plan to appeal.

Earlier cases such as Whelan v Jaslow (1986) and Lotus v Paperback Software International (1990) seemed to give strong support to arguments of this kind by placing emphasis on the similarity of user interfaces in finding that copyright had been infringed. More recently, in cases such as Computer Associates International v Altai (1992), the US courts have placed a narrower interpretation on software copyright, but it still seems to be accepted that copyright can extend to the non-literal elements of a computer program.

These US cases have parallels in the UK and other jurisdictions. The latest case in England, which was
decided in February 1993 under the old law, seems to apply a ‘look and feel’ approach to the question of infringement, although the judge is careful to break down the elements of the program into those which do and those which do not have copyright. In March 1993 the High Court of Australia reaffirmed its view that copyright protection went beyond the literal elements of a program, and sustained a judgement that a ‘look-up table’ in a hardware lock program was a substantial part of that program, the copying of which was an infringement of the program’s copyright.

From Apple or Lotus’s viewpoint, the interfaces of their systems are something in which they have invested considerable time and money. For instance, they may have employed psychologists to design an interface that is easily understood and manipulated by novice users. It is a market edge they want to retain rights over. From Microsoft, Borland or, indeed, the user’s viewpoint, some interfaces have become de facto standards which it is only sensible to follow. Users want as much standardization as possible to make it easy to learn new systems and to move between products with similar functionality. Software producers want to build enhanced systems on established standards. Resolving these conflicting demands is a political problem that has become urgent.

This political issue has generated a lot of interest. As well as media coverage a number of pressure groups have become involved. One of the most vocal has been the League for Programming Freedom (see also SAGE and ECIS in Section 3) founded by Richard Stallman from MIT, the author of the public domain GNU software. Stallman sees the extensions of both patent and copyright law as a threat to the system of free software which he has made his life’s work, and has campaigned vigorously against them. The authors became involved through their association with the Edinburgh Computing and Social Responsibility Group (ECSR), which has also campaigned on these issues.

‘Look and feel’ disputes illustrate one of the ways in which regarding programs as literary works is a fiction. It is impossible to have two books that are indistinguishable in appearance while being written in different words, but this situation can readily arise with programs. Thus ‘look and feel’ disputes cannot be settled on the basis of precedent. New law is required.

Judgements on ‘look and feel’ disputes have gone in both directions. Many of those judgements have gone on to appeal. So it is too early to predict the final outcome. It is possible that the reach of copyright law could be considerably extended. This decision is being left to judges, with little guidance from the existing legislation.

Another controversial question in the USA is the existence or otherwise of a decompilation right which users can set against the claim of the copyright owner. Two cases in 1992—Atari Games Corp v Nintendo and Sega Enterprises v Accolade—held that decompilation fell within the scope of ‘fair use’, a copyright concept allowing certain uses of works on grounds of public interest which would otherwise infringe copyright. There are equivalent provisions in UK law under the heading ‘fair dealing’, which allow limited copying for purposes of private study, research, criticism and review. However, there have been no court decisions on these points in the context of computer programs.

4. THE EUROPEAN DIRECTIVE

With the advent of the single European market, the European Community has been trying to harmonize the trading laws of members states. In 1990 it turned its attention to the different software copyright laws, which were widely disparate—where they existed at all. Its aim was to reach an agreement and issue a Directive, which member states would then implement by modifying their separate copyright laws. This Directive was issued in May 1991.

As well as harmonization, the Directive also provided the EC with the opportunity to strengthen the position of European software producers against American market dominance and to encourage the creation of standards. To counter this move and protect the market leaders, the pressure group: Software Action Group Europe (SAGE) was formed mainly from large US-based companies. SAGE influenced the Directive, strengthening the copyright protection of software producers. To reverse this influence the pressure group: European Committee for Interoperable Software (ECIS) was formed mainly from smaller, European-based companies. The resulting Directive was a compromise between the positions of the two pressure groups.

In particular, the issue of ‘look and feel’ was left unresolved to be settled by the laws of the individual member states. The Legal Affairs Committee of the Community did try to resolve it with an amendment to disallow ‘look and feel’ copyright, but this was defeated. The defeated amendment started:

‘Whereas, for the avoidance of doubt, it has to be made clear that only the expression of a computer program is protected and that ideas and principles, procedures, processes, systems, methods of operation and concepts which underlie any element of a program, including its interfaces, are not protected by copyright under this Directive; Whereas, these unprotectible items include, for example, protocols for communication, rules for exchanging or mutually using information that has been exchanged, formats for data, and the syntax and semantics ‘of a programming language,...’

However, a much less specific statement was included (Article 1.2):

‘...Ideas and principles which underlie any element of a computer program, including those which underlie its interfaces, are not protected by copyright under this Directive’

Nevertheless, the Directive took some trouble to distinguish the legitimate interfacing to the programs and hardware of a rival manufacturer from the illegitimate reverse engineering of a rival product in order to produce a cheaper clone. For instance, ‘decompilation’ of a program is allowed where this is ‘necessary to achieve the interoperability of an independently created program with other programs’ (Article 6.1). Such measures were
clearly aimed at helping the smaller (often European) producer to build on the products of the larger (often American) rival, but without discouraging investment in research and development.

5. THE UK REGULATIONS

The European Directive required member states to incorporate its provisions into their own laws by 1st January 1993. In August 1992 the UK Patent Office circulated a draft of the proposed UK law: ‘The Copyright (Computer Programs) Regulations 1992’. This draft was very short—only 8 pages of double-spaced A4. Unfortunately, this did not make them easy to read as they consisted mainly of detailed amendments to the earlier ‘Copyright, Designs and Patents Act 1988’, which is over 300 pages long and must be studied in parallel.

5.1. Criticisms of the draft regulations

The Edinburgh Computing and Social Responsibility Group obtained a copy of the draft Regulations and prepared a commentary on it, which we sent to the Patent Office in September 1992. As well as some minor points concerned with improving the drafting of the Regulations, ECSR made two major criticisms.

1. The draft Regulations did not address the interface issue at all. ECSR argued that it was required to by Article 1.2 of the European Directive, i.e. that it was required to mention the exclusion of ‘ideas and principles ... including those which underlie its interface’ as quoted in Section 3 above.

2. The right to decompile programs for research or private study was removed. ECSR argued that this would seriously damage the legitimate interests of researchers, teachers and students and that this removal was in violation of Article 5.3 of the European Directive.

This first criticism has been ignored. The final version of the Regulations makes no reference to ‘interfaces’. This omission came as no surprise, since Lord Reay, the minister responsible, wrote to one of us (AB) in July 1991 that ‘[the exclusion of the ideas and principles underlying an interface] is already the case in UK copyright law’ so that ‘We have therefore no intention of amending UK legislation in this respect’. However, Lord Reay accepted that some US courts ‘have taken a very broad view as to what constitutes an infringement of the copyright in a computer programme’. He had ‘no reason to believe that UK courts would follow that reasoning’. (A view, which in the light of the English February 1993 case mentioned in Section 2, now appears to be mistaken.) ECSR had argued that it was better to make the law explicit on this point than to rely on hope, but the UK Government rejected the opportunity to do this afforded by the Regulations. We must now wait to see what attitude the US and UK courts finally take on ‘look and feel’ disputes.

ECSR’s second criticism seems to have been addressed. Regulation 11 now states:

‘Where a person has the use of a computer program under an agreement, any term or condition in the agreement shall be void in so far as it purports to prohibit or restrict—

(c) the use of any device or means to observe, study or test the functioning of the program in order to understand the ideas and principles which underlie any element of the program.’

This appears to permit the decompilation of programs for purely research and study purposes, provided you have the right to use that program.

Also some of ECSR’s minor criticisms have been addressed, e.g. a very confusing and too broad definition of ‘decompilation’ has been dropped. So too has an apparently inadvertent ban on the compilation of source code.

5.2. New rights for users

The largest part of the new law is Regulation 8, which makes explicit the rights of lawful users of programs. These include the rights:

1. To make any necessary back-up copy.
2. To decompile the program in order to create a new program which can interface with the program decompiled or another program.
3. To correct errors in the program.

Item 2 above, in particular, can be seen as a victory for the ECIS position in the European debate. It should make it easier for new software producers to break into the existing market by enhancing existing successful products by interfacing to them.

However, these new rights are hemmed in with conditions. In particular, any information gained in the study of a program must not be used to create a substantially similar program nor passed to a third party without legitimate reason. These conditions are intended to meet the objections of SAGE to the use of reverse engineering to produce cheap clones.

Regulation 8 leaves lots of room for interesting legal disputes. How similar is ‘substantially’? Was a piece of information gained by decompilation or was it common knowledge? What exactly is ‘decompilation’? When is it ‘necessary’ to make a back-up copy? How many back-up copies is it reasonable to make? As usual, the lawyers are likely to earn some money.

6. PROGRAMS AS LITERARY WORKS

The European Directive and the UK Regulations were needed because computer programs did not quite fit existing copyright law. Defining programs as literary works is convenient because it avoids the need for lengthy, and possibly unsuccessful, international negotiations. However, the analogy is not exact.

Unlike literary works, computer programs can be described at a number of levels: from machine code
through high-level languages to the interface they present to the user. Two programs can be similar at higher levels but differ at lower ones, e.g. similar interfaces—different code; similar source code—different object code. To which level(s) does copyright apply—or, putting it another way, to which levels does copyright not apply?

- Copyright applies to the 'expression' of a work. This seems to be particularly difficult to define for computer programs. On the one hand it must be abstract enough to cover different representation media (discs, tapes, paper, etc.) and different programming languages. On the other hand it must not be so abstract as an algorithm, which is a mathematical concept which cannot be copyrighted. There seems to be no technical concept in computer science which corresponds to the legal notion of 'expression'.

- Copyright law usually assumes the author of a work to be a natural person or group of natural people (see Article 2.1 of the European Directive). What about works produced by machine? Machine production of regular literary works is possible but rare. However, machine production of object code is standard and machine production of source code is a growing research area. The UK is the first country to try to deal with computer generated work. Unfortunately, it does so in a rather vague and obscure way. Section 4(3) of the UK Copyright, Designs and Patents Act 1988 says:

> 'In the case of a literary, dramatic, musical or artistic work which is computer-generated, the author shall be the person by whom the arrangements necessary for the creation of the work are undertaken.'

The Government rejected the advice of ECSR to remove this section, but they will eventually have to do something about it.

- Literary works are passive objects, but computer programs are active; when run they have effects. They can be implemented either in software or in hardware. Of course, these are the reasons that patent law has also started to be applied to software techniques in a way that they could not be applied to literary techniques.

Despite these problems a strong consensus is emerging that adapting the existing copyright law is the best way to deal with computer programs. Following the European Directive, the new GATT TRIPS agreement proposes that all signatory countries should apply the Berne Convention to both programs and databases.

We can only hope that these latest amendments to UK copyright law have found a way to maintain the fiction of computer programs as literary works. Time will tell.