Copyright and information technology

In the 1960s, computer programs were considered accessories to the very expensive computers. There were several reasons for this, one that a program simply could not be run on any other computer than the one for which it was written, high-level languages were still in the making\(^1\) and compatibility was low. But IBM had considerable success with its 360-series announced in 1964 allowing customers to purchase a smaller system and migrate upward if their needs grew. In 1969 IBM decided – perhaps somewhat stimulated by the anti-trust suit to which it was party\(^2\) – to unbundle hard- and software. As computer programs were separately priced, it became possible for third parties to offer competing programs. And in such a market arose the obvious issue of the protection of computer programs.

At this time, it was still unclear to what extent the US Copyright law applied to computer programs. There were several court decisions, the copyright and patent systems competing to become the legal framework for the intellectual property protection of computer programs. There were also strong advocates for a third possibility, a *sui generis* regime for computer programs, as it was pointed out that neither copyright nor patent was designed to accommodate the special features of computer programs, the characteristic which is occasionally called ‘industrial copyright’, programs themselves often called ‘soft machines’.

The author will be permitted an anecdote\(^3\) by way of illustration. At one of the meetings of experts\(^4\) to the WIPO in Geneva,\(^5\) there had been an unusually heavy snowfall during the night. Struggling uphill to the WIPO building, one

---

1 The first version of COBOL was adopted in 1968 by the American National Standards Institute.
2 The complaint for the case *US v. IBM* was filed in US District Court, Southern District of New York on 17 January 1969 by the Justice Department. The case was withdrawn on 8 January 1982.
3 This is based on my own notes from the meeting.
4 Advisory Group of Governmental Experts on the Protection of Computer Programs.
could see improvised tools being used to remove the snow in order for parked cars to escape. At the meeting, the head of the delegation of the Soviet Union\(^6\) made this the basis of a metaphor pleading for a \textit{sui generis} solution, ‘In Geneva, where the snow rarely falls, one may allow oneself to adapt the tools at hand for the removal of snow. If you live in Moscow, you will expect the snow to fall heavily every winter, and you will have efficient and specialised tools. And I ask you, ladies and gentlemen, do you think computer programs are like the snow in Moscow or in Geneva?’\(^7\)

WIPO actually developed the 1971–77 ‘Model provisions on the protection of computer software’ with the assistance of Professor Peter Seipel,\(^8\) but these were not adopted as national legislation in any country. The model provisions were inspired by copyright, but had some elements akin to patent protection of the content of programs. In practice the discussion of alternatives came to a halt when the US adopted the 1980 amendments to the 1976 Copyright Act, extending copyright protection to computer programs. A country was free under the conventions to qualify programs as literary works, and this made it possible nearly overnight to establish an international scheme of protection, based on the Berne and Universal Copyright Conventions.

The interest in copyright was nearly exclusively limited to computer programs. For these there was a market, and there were strong commercial interests in protecting programs. This interest also found different strategies for protection; one was to introduce various devices which had to be present for the program to be executed, like an extra element for the serial plug to the printer which then was called by the program, which failed to initiate printing if the element was not found. This was the beginning of technical protection measures, the discussion of which later escalated. Another obvious measure was only to make the program available in object form, which in turn gave rise to the doctrine of and provisions on reverse engineering in order to make it possible to develop programs functionally interacting with another program. A characteristic of copyright is that the protection allows anyone to access the information in the protected work, and use this information in the creation of new and independent works. The practice of making programs available in object form only, barred access to the information, and reverse engineering may be seen as a reaction to this, for copyright, somewhat alien aspect.

---

\(^6\) Vitaly Troussov, Deputy Director of the Patent Examination Department, USSR State Committee for Inventions and Discoveries.

\(^7\) One will find a reference to this intervention, though stripped of the images, in the report of the meeting, paragraph 22, UNESCO/WIPO/GE/CSS/3, 8 March 1985, 4.

\(^8\) Swedish Law and Informatics Research Institute, Faculty of Law, University of Stockholm.
There was some interest in other aspects. A joint WIPO and UNESCO meeting\(^9\) of 1982 concerned the ‘problems arising from the use of computer systems for access to or creation of works’. In the recommendation it is stated that for instance uploading a protected work to a computerised system represented a reproduction in the terms of the conventions. The use of computers to create work attracted some attention. At this time composers would use computer programs as tools, and the recommendation\(^10\) also states that ‘tool’ is the perspective in which to consider such use.

However, there were considerable limitations in computerised systems at this time (1982) for a real concern about the use of literary, musical or audiovisual works to emerge for computerised systems. The IBM PC had been brought out the year before. The first model did not have a hard disk, but only 5 1/4 inch floppy disks (and they really were floppy). Storage was still expensive. Only with low storage costs could the volumes of data involved for storing protected works be considered. In the early 1980s, the emphasis was on programs and the special type of programs used for gaming in the first low cost specially designed consoles brought out for the lower end of the consumer market. Also, infrastructure had to develop for the establishment of a market for protected works. This did not happen until the early 1990s. These developments shaped the Web, and at the same time created the potential for a market in protected works for which legal policies are still unfolding, and which promise an interesting future for the law of intellectual property related to information technology.

### Computer programs

**Source and object programs**

The term ‘computer program’ is part of our common language. Like other such terms, it is used on the basis of linguistic conventions of everyday technical language, and consequently is somewhat vague. At the core of the concept is the notion of a program which can be executed by a computer. The program will then have to take the form of a set of instructions conforming to the formalism of a programming language. The language consists of a certain predefined *set of commands*, which semantic is defined in detail with respect to the actions they will cause in a computer. The language will also have a defined *syntax* which has to be followed strictly for the program to be executed, or executed correctly. The program will permit variables which may be chosen by the programmer with considerable freedom. The program is seen

---

\(^9\) UNESCO is the depositary for the Universal Copyright Convention.

\(^10\) Art. 14.
as different from the *data*, which is what the program operates upon, that is, the relation between a word processing program and the texts processed by that program.

In simple terms, a diagram may indicate some of the different aspects of a program (see Figure 16.1).

The notion of a computer program is centred on the version in *source code*. This is a program written in a high-level language. This may be one of the traditional languages like COBOL, but is more likely to be a modern object-oriented language like C++ or RUBY. In our context, this is not very important. Such a programming language is often characterised as a high-level language, implying that it is ‘high’ in contrast to the programs existing at ‘deeper’ levels in the computer – the deepest level is where the program instructions are broken down to primitives which correspond to the machine components, ultimately the transistors, accumulators etc. which are present in their millions on the integrated circuits of the computer. The programming language exploits these possibilities to carve out hunks with special functions. Such a hunk is then given the name of an instruction, and using this instruction – which may be the word SORT – the programmer invokes the whole hunk of these special functions. The high-level programming language can be read by a human who is familiar with the language, in much the same way as we learn to read algebra. There are elements in the high-level language which are similar to natural language, like the word SORT, and its ‘meaning’ is related to the meaning of the word in natural language. But in contrast to natural language, the programming language has a detailed definition of its semantic (or understanding).

Typically, the program in high-level language will be run through another program to replace the high-level instructions by the machine-related detailed

![Diagram](figure16_1.png)

*Figure 16.1     Introducing computer programs*
instructions necessary to run the computer. This program is called a compiler. In this the simple sentences of high-level language are exploded into a very high number of primitive instructions implied by the definitions. The result is the object program, which may govern the functions of a computer. The compiler also checks the logic of the object program, and will report errors to the programmer. What may above have appeared as a one-way stream is really an interactive process, where the programmer has the program compiled as he or she works on developing the program, using the feedback of the compiler to improve on the program.

There is a one-to-one relationship between the source and object versions of the program in the sense that if two identical source programs were compiled by the same compiler, the result would be identical object programs. Occasionally one sees the process described as a ‘translation’ from object to source versions. This is not an appropriate choice of language, as translation conventionally is understood as a situation where the translator has a certain freedom of choice with respect to picking the right words or phrases. No such freedom exists for the compiler, it is a process governed by strict rules. It may be better compared to a machine which converts a text of Latin characters into Morse code, each letter being replaced by a pattern of dashes and dots.

There also is a one-to-one relationship between object and source code. Running the object code the reverse direction through a re-compiler, a source code will be produced – and running the same object code through the same re-compiler will each time result in an identical source version. But there is no one-to-one version between the original object code and the re-compiled object code. In compiling, a number of predefined choices are made. Several source code constructions may result in an identical set of object code instructions. And the compiler will make the object code optimal for the performance of the processing. Re-compiling the object code, there will be many possibilities for generating source code structures. While, originally, the programmer organised the source program for easy use by humans, the re-compiled program will reflect a structure made optimal for computer processing, and therefore pose challenges for the human sitting down to understand the re-compiled source.

In practice, the relation between source and object versions does not pose many difficulties for copyright law.

**Preparatory design work**

When starting to develop a program, someone must have an idea or take the initiative. There is no shortage of examples of the same person having a bright idea and developing it into a successful source program. But often the situation is more mundane – an organisation has a need for a certain function, and
requests a program to be developed to satisfy this need. Before the proper programming starts, there will be material describing the need, suggesting solutions etc. In Figure 16.1, this is indicated by the boxes ‘design material’ and ‘specification’. These obviously cannot be compiled, and cannot result in a set of instructions operating a computer. But according to the terminology of the Computer Program Directive, they are to be understood as ‘computer programs’ (see the preamble item 7):

\[\ldots\] whereas this term also includes preparatory design work leading to the development of a computer program provided that the nature of the preparatory work is such that a computer program can result from it as a later stage.

The criterion to distinguish preparatory work which is just an ordinary literary work of non-fiction discussing the development of a computer program, and preparatory work which is protected as a computer program, is whether a computer program ‘can result from it as a later stage’. This is certainly not a very clear criterion. It must obviously cover a situation in which a program has been specified by a formalism – for instance quasi-coding – which leaves little freedom for a programmer in transforming it into a source program. But how much freedom should be allowed before there is an independent literary work and an independent program has to be decided in the context of a concrete case.

One should keep in mind that there may be a case of joint authorship. The programmer may have sufficient freedom to imprint the resulting program with his or her choices, but at the same time the preparatory work governs the programming in such a way that it becomes a derivative work: the program is not independent of the preparatory literary work. This may be appropriate also because there is often a functional division between the person designing the program, resulting in a high-level specification, and the person coding the program on this basis. The designer may very well be the person with the original bright ideas, indicating functions and their interrelation, while the programming may be more of a sweat-of-the-brow task realising the indicated functions. In the abstract, this relation between designer and programmer, corresponding to the issue of when a program is an adaptation of a specification, may seem difficult. In practice, the issue will generally be solved by the rights of all those involved in the project being collected in the hand of the employer or through contracts governing the project (see below).

One may observe that the inclusion of preparatory design work makes it

explicit that computer programs do not have to be machine readable in order to fall within the scope of the Computer Program Directive. In principle, a source program printed as an example in a textbook, will be a program under the Directive, though this may create some detailed problems in practice.\(^{12}\)

**Originality**

Article 1(1) of the Computer Programs Directive specifies that computer programs are to be protected as ‘literary works within the meaning of the Berne Convention’. This implies that computer programs are to be treated as other works within the category of ‘literary works’, typical examples of which are novels or textbooks. This would seem appropriate. A source program has the appearance of a text when printed out, though generally more structured and repetitive in the choice of terms than a piece of natural language text. However, the main difference is that the text of a computer program through the process described above is transformed into instructions governing a computer – it has a *functional* aspect not found in other literary works. Even a very technical text relying on algebra, or a technical drawing defining a machine in detail, does not have functions – there is no way such a work can be made to execute any process in the real world by itself. The functional aspect is therefore a fundamental difference between computer programs and other sub-categories of literary works, and this may have consequences for the interpretation of the Directive or national law in some situations.

The criterion for achieving protection is set out in article 1(3) of the Computer Programs Directive:

> A computer program shall be protected if it is original in the sense that it is the author’s own intellectual creation. No other criteria shall be applied to determine its eligibility for protection.

This is an attempt to harmonise – at least with respect to computer programs – how to decide that a computer program rises above the threshold necessary to be qualified as a copyrighted ‘work’. The Berne Convention does not itself have any definition or qualification of the term ‘work’, though guidance may be sought in the exemplification of article 2(1) of the Berne Convention. The criterion seems to require more than observing that the program is not copied from an earlier program (that it is ‘new’), as it has to be the creation of its author. How much should be added to the criterion ‘not copied’ is not very clear. But in its report on the Computer Programs Directive, the Commission

\(^{12}\) See infra.
has stated that twelve member states were required to lower, and three to increase, the threshold.\textsuperscript{13}

The major decision signalling that the criterion of originality was interpreted as more severe with respect to computer programs was by the German Supreme Court.\textsuperscript{14} A decision by the same court, subsequent to the adoption of the Directive, confirmed that the threshold had been adjusted.\textsuperscript{15} One of the jurisdictions which had to raise its threshold was the United Kingdom, where a ‘sweat-of-the-brow’ doctrine would seem to have been accepted.

In practice it may be quite difficult to appreciate the originality of a computer program. Often a program is composed by using third-party components which are ‘glued’ together by simple coding. The contribution of the author’s own intellect may be how to do this, and the result may be similar to an anthology, to which the creative editor may have copyright. In practice, the problem does not arise too often – and there seems to be considerable guidance in the rule of the thumb: if a computer program has market value, it is also a copyrighted work.\textsuperscript{16}

In this introduction, computer programs have been presented as instructions and the preparatory design work which results in programs. But a ‘program’ will also have other elements. The code may represent graphical interfaces which may be protected as technical drawings or other examples of art. The program may rely on feedback messages contained in separate libraries of different languages for user communication. And there may be integrated systems for help and assistance. To the user, this is all one ‘package’. Generally, such elements are not considered in copyright terms to be part of the literary work qualified as a program (though there may be modifications; see below under reverse engineering). And such elements may be subject to protection under copyright law or related rights (for instance, the indexes or help files may be qualified as protected databases).

In general, one should be aware that a ‘program’ is a system concept. One program may consist of sub-programs, and interact with other programs to constitute one functional service. To decide what is one program will not be a technical issue alone; it also has to take into consideration the marketing context, what is presented as ‘a program’ to users, etc.

\textsuperscript{13} Thomas Dreier and Bernt Hugenholtz (eds), \textit{Concise European Copyright Law}, Kluwer, Alphen aan den Rijn 2006, 217. The express objective of harmonisation is stated in the Preamble item 4.


\textsuperscript{16} Formulated by Mogens Koktvedgaard, Danish professor of intellectual property law, refusing to be confused by subtle arguments.
The right holder

The original copyright holder: vertical, horizontal and temporal cooperation

The author of a computer program is:

... the natural person or group of natural persons who has created the program or, where the legislation of the Member State permits, the legal person designated as the right holder by that legislation. Where collective works are recognized by the legislation of a Member State, the person considered by the legislation of the Member State to have created the work shall be deemed to be its author.

This is no different for computer programs than for other literary works. In many instances, there is no problem in readily identifying the author, who is the person conceiving and coding the program, using his or her exclusive rights as a platform from which to negotiate the exploitation of the program.

However, computer programs are part of ‘industrial’ copyright, and often the context will be different from that in which the traditional literary works are created. It may be somewhat similar to the development of textbooks, which may be huge projects initiated by a publishing house involving several authors, illustrators and currently also web designers and possibly programmers. The differences from the traditional context may be summed up as follows:

- the initiative is usually taken by an institution or a more loosely organised ‘project’;
- there are usually created specifications for the coding of the program;
- there are usually several physical persons involved in coding the program;
- third party elements are occasionally purchased to be integrated with the program;
- the program is developed to be present in the market for some time (years);
- the program will be maintained over time by making adjustments, corrections, or adaptations.

In some jurisdictions, legal persons may be the original copyright holder. According to the Continental European model, the original copyright holder must be a physical person, and Figure 16.2 below illustrates aspects of this situation.

The coding of a program may be carried out by several, even many, persons. One may have large projects where the program is divided into parts with well-defined interfaces. Groups may have responsibility for one module, working in parallel with other groups under a coordinating project management. The result will be a work ‘created by a group of natural persons jointly’,...
and therefore ‘the exclusive rights shall be owned jointly’, see article 2(2) of the Computer Programs Directive. But this is only one of the several situations where such joint creation may take place; also indicated are the possibility for ‘vertical’ and ‘temporal’ cooperation.

Here there is ‘vertical cooperation’ between designers, specifying the program, and programmers, coding the program. This relationship has been briefly commented on above.

A prominent feature of computer programs is their maintenance. The use of programs will generate experiences which will be fed back to the design of the program. In any program of a certain complexity there will in practice always be ‘errors’ in the form of logical inconsistency. There will always be the possibility of improving performance. Throughout the life of a program, there will be new programs and hardware gadgets being introduced with which the program should interoperate, which will require further development of the program. And there may be functional enhancement. This is simplified in Figure 16.2 as a feedback loop amending the specifications of the program, requiring further programming. Errors may be addressed by the release of a patch to be integrated with the program. Enhancement may result in a new ‘version’ being released. Over time the program will be amended, perhaps even adapted. This change may be incremental; over time the program will change – and this change will be brought about by different persons than those involved in the first development.

The result is a ‘joint work’. As illustrated, this may include many physical individuals, and the relation between the work and a person who has contributed with his or her intellectual creative force to the program, may be rather tenuous. Some form of rights management is necessary, and this is

---

**Figure 16.2  Cooperation in developing a program**
usually supplied by the institution, project or other entity which has taken the initiative in the programming. There will therefore be an assignment of rights from the original copyright holders to the institution in question, and this institution will then face the market with a whole bundle of rights as a platform for negotiating further contracts.

One will note this initial rights management is rather critical. If one of the original copyright holders fails to transfer his or her rights, there will be a defect in the title to the overall program. In practice this may happen. Often an implicit assignment is implied by the context in which the work is carried out, and there may be few possibilities for such a person to actually to exploit the situation in a commercial way. These practical circumstances are probably the reason for there being few examples of conflict on this basis.

**Employee’s assignment of copyright to employer**

When organising the development of a computer program, an obvious way to organise the project is through employment of designers, programmers and others. There will then be an employment contract between the employer and the employee.

If a person is employed to create works – this is for traditional categories of jobs the case for journalists, both in paper and ether media – the rights in the created work obviously have to be shared in some way with the employer. It is understood that in this respect, the law varies between European countries. In some jurisdictions, the employer (also when a legal person) may be the original copyright holder. In other jurisdictions, the original copyright holder is always a physical person – the employee – and also the relationship between the employer and employee is seen as subject to contractual arrangements and in the perspective of labour law rather than copyright law. In the Nordic countries, journalists’ organisations would maintain it is not appropriate for the legislator to interfere with the right of the unions to enter into collective agreements, including with respect to the transfer of copyright between employee and employer.

Therefore, there is not a unified background law in this respect. In spite of this, article 2(3) of the Directive on Computer Programs harmonises the transfer of employees’ rights with respect to programs:

> Where a computer program is created by an employee in the execution of his duties or following the instructions given by his employer, the employer exclusively shall be entitled to exercise all economic rights in the program so created, unless otherwise provided by contract.

In this instance, European law has been harmonised. But it is a very limited coordination. As indicated above, the background law on transfer between employee and employer varies between jurisdictions. Also where there are
contractual arrangements, there may be different rules for interpretation, often in favour of the original copyright holder (who in the traditional view is the ‘weaker’ party in the relationship).

The term ‘computer program’ has to be interpreted according to the Directive, but may not be applied analogously for other types of work. Against the background of a default legislation based on the ‘hands off’ policy indicated above, the interpretation may be rather restrictive, for instance not applying to the design of databases. The term ‘employee’ is also a bit of a hurdle; in internal law there may be different definitions of ‘employee’ with respect to social security law, working environment law, tax law etc., especially the qualification of ‘employee’ with respect to a ‘consultant’ may be open to interpretation, as in projects for developing a program there are often inventive ways of associating individuals with a project. Obviously there may also be difficulties in determining whether the efforts are contained in the duties of the employment contract, or whether the employee himself or herself could have taken the initiative to develop the program.

Without elaborating this further, it is obvious that national law may vary in this respect. But these considerations on possibilities of divergences in national law should not overshadow the principle, which is also appropriate as a guideline for managing the rights to a computer program: the establishment of a practical regime of rights management which implies the collection of the many rights in one hand, with appropriate upstream contractual arrangements.

The exclusive acts
Having decided what is subject to the exclusive right (a computer program) and who is the right holder, one may approach the more succulent part of the issue – the exclusive acts reserved for the right holders.

The right of reproduction

The more important exclusive right is the right to reproduction, stated in article 4(a) of the Directive:

the permanent or temporary reproduction of a computer program by any means and in any form, in part or in whole. Insofar as loading, displaying, running, transmission or storage of the computer program necessitate such reproduction, such acts shall be subject to authorization by the right holder . . .

‘Reproduction’ is interpreted according to the Berne Convention. But this does not have any definition of reproduction. The Directive includes temporary reproductions, and this was for some time an issue. It is believed that this
issue is resolved by article 2 of the Copyright Directive,\textsuperscript{17} though this does not directly apply to computer programs.\textsuperscript{18} The discussion leading to the adoption of the Copyright Directive must be taken into consideration, and the Directive as the conclusion of the discussion, which may be seen as addressing the permanence necessary for a representation to qualify as a reproduction. It is rather obvious that a ‘permanent’ reproduction on paper, magnetic or optical medium is a reproduction. It is also obvious that there are passing representations which are not qualified as reproductions, like the reflection of a painting in a mirror.

Processing a work by a computer implies numerous instances of copying. In order to examine a text on a computer screen, the text has to be communicated from a storage medium, typically a magnetic disk, to the central processing unit of the computer, where the representation is used to control the properties of the screen so that an image of the text appears. This representation in the CPU is short-lived, the storage cache has a limited capacity (but may in modern systems be of several gigabytes), and will be overwritten by the operating system if necessary. Also, data will be paged in and out of the CPU using techniques for virtual memory to make the user experience that a larger volume of data is available from the cache than in fact is the case. Similar solutions are used in communicating data through a network, where packets may be stored intermediately in nodes of the network waiting for forwarding capacity; the time stored varying upwards from microseconds.

There was disagreement as to whether these short-lived representations qualified as copies. The Copyright Directive would seem to have put an end to that disagreement, and the Computer Programs Directive uses the same terms in defining a reproduction. It must therefore be held that all these examples qualify as reproductions. In practice, there therefore would seem to be (at least) three degrees of permanence of a representation: those too volatile to qualify as a reproduction (like a mirror image), the temporary reproductions in computer-based systems, and permanent reproductions.

The author’s own view on this issue is probably visible even through the brief discussion above. It is believed that the issue originally was caused by some concern with time-sharing computer bureaux. Time-sharing was popular in the 1980s. Several users would share a mainframe, accessing the computer from (dumb) terminals. A user would typically have an account and a password, and would have access to a segment of the mass storage, where the programs licensed by the user were available, and where data could be stored.


\textsuperscript{18} See art. 1(2)(a), Copyright Directive.
The scenario suggested a disloyal user accessing the mainframe, using the password of another user. In this way, the disloyal user could process data stored on his or her segment of the mass storage device by a program licensed by the user from whom the password was obtained. This would then not constitute a copyright infringement as the program was not reproduced – unless one qualified the temporary representation in the CPU of the terminal employed by the disloyal user as a reproduction. Otherwise, one would have to refer to the criminal law on hacking, which at that time was not very far advanced or coordinated between jurisdictions.

The Computer Programs Directive specifies that ‘such reproduction’ necessary for ‘loading, displaying, running, transmission or storage’ of the program is part of the exclusive right, which includes typical examples of permanent reproduction (‘storage’) and temporary reproductions in the meaning indicated above (‘loading, displaying, running, transmission’). The result is a very strong exclusive right of reproduction.

In addition, the Computer Program Directive extends the exclusive right through article 4(b) to

... the translation, adaptation, arrangement and any other alteration of a computer program and the reproduction of the results thereof, without prejudice to the rights of the person who alters the program ...

This makes it explicit that amendments or derivative works are also included in the exclusive right. Perhaps the use of the term ‘translation’ deserves a comment: translation from one high-level programming language to another is not necessarily identical to translating a text from one natural language to another. Programming languages may be related, or they may be based on widely different design principles. If a program is written in a logic language like PROLOG, the process of representing the rules in a language like COBOL would certainly be more than a mere translation, and perhaps better seen as taking a detailed specification to develop a new program. As algorithms, methods etc. are not protected, one may in such a case argue that there is no relevant derivative relationship between the programs. This is just an observation warning that one will have to look at what actually has been done rather than the term used to describe the process in order to decide whether it is a reproduction.

DELIMITATION FOR REPRODUCTION FOR INTENDED USE

It is obvious that the resulting exclusive right to reproduction is so strong that it has to be modified in order to make the market work in practice. Therefore, article 5(1) of the Computer Programs Directive makes a delimitation of the exclusive right for reproduction:
In the absence of specific contractual provisions, the acts referred to in Article 4 (a) and (b) shall not require authorization by the rightholder where they are necessary for the use of the computer program by the lawful acquirer in accordance with its intended purpose, including for error correction.

One will note that the provision is secondary to a deviating contractual regulation, and therefore presume that there is such a relation between the parties. In this context, there is no opportunity to further discuss the practices of contracting for computer programs. Of course, these vary from the detailed and complex documents governing the major development of a program system to the transactions across the counter, where no negotiation takes place, and the terms will flow from background law. It may not be unfair to maintain that the Computer Programs Directive is drafted on the understanding that the transaction will involve the acquisition of a physical medium, typically diskettes or compact disks, on which the program is stored. At the time the Directive was drafted, it was not uncommon that a program also was executed from a diskette inserted in the station of the computer. Development made this obsolete; the medium is only used for transporting the program to the computer on which it is to be used – here a reproduction takes place uploading the program to the internal memory of the computer. This reproduction is permitted by article 5(1) of the Computer Program directive, as it is in accordance with the intended purpose of the program. When executed, temporary copies will be made in the CPU; these also are permitted as they are in accordance with the intended purpose. But without contractual permission, only one such copy can be made of the program at the same time. It is the original copy on the physical medium which is the copy acquired, and it is this that governs the use of the program – the uploading to a computer is permitted because it is required for using the original purchased copy according to its purpose. Therefore it also must be permitted to delete an uploaded program and re-install the program on another computer; the relation between the original copy and the reproduction remains. There may be a certain leeway found in ‘intended purpose’, but the right to reproduce the program according to the Computer Program Directive article 5(1) remains rather narrow.

As emphasised, the delimitation of the Computer Program Directive article 5 rests on the presumption of a contractual relationship between the parties. Nevertheless, in implementing the provision in national law, at least two different strategies have been adopted. One is what would seem to follow the presumption of the Directive, introducing a clause governing the interpretation of licences for computer programs – many national acts will already have clauses for special categories of contracts, for instance the publishing contract or contracts for making a cinematographic work. However, another strategy is to introduce the substance of the Computer Program Directive as delimitations
of the general exclusive right of a right holder independent of any contractual relationship, but subject to such contracts in the sense that a contract will be given priority when the legislation does not state otherwise. The result is in practice rather similar, but in certain special situations there may be relevant differences (see below).

BACK-UP COPIES
As anybody dependent on a computer system will know from their own experience, there will occasionally be situations in which the data stored on local drives are lost. A basic principle of data security is that back-up copies should be made. The Computer Programs Directive permits reproduction for this purpose (see article 5(2)): ‘The making of a back-up copy by a person having a right to use the computer program may not be prevented by contract insofar as it is necessary for that use.’

The importance placed on this principle is evident partly from this being a mandatory right. Reproduction for this purpose may be done by ‘a person having the right to use the computer program’, which will include a somewhat wider circle than those having the right of reproduction for the intended use, which is limited to ‘the lawful acquirer’. The right to make a back-up copy will certainly include technical service staff routinely maintaining a computer system.

It may not be obvious why one needs a back-up copy when the program is acquired on a separate medium. If the program is lost on the computer on which it has been installed, a new installation may be made from the original copy, as happened in the first place – the original copy filling the need for a back-up. But a user of a program will typically make many choices in setting up the program to fit his or her preferences. All such features would be lost if a back-up copy could not be made.19

The right to make a back-up copy is limited to computer programs, a concept discussed above. In what for the user is perceived as a program, there may be elements which are not a computer program according to a strict inter-

---

19 UK courts have held that back-ups may not be necessary when the program is made available on a CD-ROM, see *Sony Computer Entertainment Inc v. Paul Owen and others*, [2002] EWHC 45 (ChD). The decision concerns use of a device to circumvent the geographical segmentation imposed by Sony on game modules, but the opinion of Justice Jacob on this point is given in very general terms. A similar view is taken by Justice Laddie in *Kabushiki Kaisha Sony Computer Entertainment Inc et al v. Ball et al*, [2004] EWHC 1738 (ChD). However, the decisions should be interpreted in the context of the disputes. A gaming module does not offer the type of user settings which in the text is indicated as creating a need for back-up copies, including also where the program is made available on a separate medium like a compact disk.
pretation of the term. A typical example would be manuals, collections of clip-art etc. The program is also typically available to the user in object form. There is no practical way in which the user can qualify which part of the object code is a computer program in the strict sense, and which elements are accessories of a different nature. The clause must therefore be interpreted to permit reproduction for back-up purposes of all files presenting themselves as part of the purchased program.

This point gains some extra interest because the Copyright Directive seems to lack a corresponding provision. It is in some way remarkable that a limitation found sufficiently vital with respect to computer programs to make it mandatory, is not even mentioned with respect to other types of work, where it will be just as vital if these works are in computerised form. For such works, one therefore has to rely on the contract to authorise the reproduction necessary for back-up routines.20

---

20 This point will not be pursued, but there are obvious tensions between the two directives at this point.
Article 5(2) of the Computer Program Directive uses the phrase ‘a back-up copy’. This seems to imply that only one back-up copy is permitted at any time. According to routines, back-ups are made at intervals – every day, week etc. If only one back-up copy is permitted, this implies that earlier copies will have to be deleted when a new is made. Again, this would be contrary to basic data security principles, which advise at any one time having several generations of back-up copies, preferably stored at different locations to reduce the risk of accidental loss.

It must be justified to interpret the clause as a reference to normal back-up routines, permitting the reproduction of computer programs to the extent this is compliant with normal routines and loyal to the right holder.

A special situation occurs when a licence to a program lapses. The situation may be that a business has contracted for a program for some vital functions of its operation, perhaps as trivial – but necessary – as accounting. A special licence agreement is negotiated which includes a maintenance arrangement, the program being made available for an annual licence fee. Variations of such arrangements are rather common for large programs requiring user adaptation. As time goes on, there emerge competing programs, and the user decides to discontinue the relationship with the first provider, and switch to another solution. The licence agreement for the first program will lapse. Without such agreement, it may be argued that no right to maintain reproductions of the program remains. This would place the user in a very uncomfortable situation, as the program is needed to access historical records. Such access would be required by law in most jurisdictions for accounting records. One may look to article 5(2) of the Computer Programs Directive and argue that the right to make and retain a back-up copy also extends to a person who had a right to use the program when necessary to access the data processed by the program in the period the contractual arrangement lasted.

REPRODUCTION FOR PRIVATE USE

According to article 5(2)(b) of the Copyright Directive, one of the possibilities for limiting the exclusive right to reproduction by national legislation is ‘reproductions . . . made by a natural person for private use’. Considering the Computer Programs Directive, one will not find a similar possibility for delimiting the exclusive right to reproduction. This implies that the Directive bars the possibility in national legislation to open for ‘private reproduction’ of computer programs.

When the Computer Programs Directive was adopted, this was a rather controversial feature. The justification obviously was the argument that computer programs were very vulnerable to reproduction, and that the limitation of an exclusive right to private reproduction would represent a substantial reduction in the exclusive right enjoyed by the right holder, and would fail to
meet the three-step test of article 9(2) of the Berne Convention. The concern was partly that individuals would copy programs from their workplace to private portable computers or home computers, but also the market for gaming modules, where the major market was private and the typical user (at least at this time) juveniles.

When the Copyright Directive was passed, the provision in general on private reproduction was made subject to a presumption of fair remuneration to the right holder. But the regime established for computer programs was maintained, with respect to this type of literary works. There is no delimitation of the exclusive right for private use. Some national implementation laws made it explicit that this only holds for computer programs in machine readable form, otherwise a program printed in a textbook would be subject to different provisions on reproduction than the rest of the book.

**ANALYSIS OF PROGRAMS**

The Computer Programs Directive emphasises the dichotomy between the expression (which is protected by copyright law) and the ideas of a work (which are not protected). This is mentioned in the preamble, and it is repeated in article 1(2), mentioning explicitly that ‘ideas and principles’ are not protected. And article 5(3) of the Directive restates this in the context that a person having the right to use a program may examine it to ‘determine the ideas and principles which underlie any element of the program’. It would seem that this is overstated in the Directive, as it is a delimitation flowing also from basic copyright law, and would apply without any direct regulation in the Directive. Perhaps the reason for emphasising the point is the strong exclusive right for reproduction given to the right holder, and the relation to the right of decompilation (see below).

*The right of distribution*

**THE EXCLUSIVE RIGHT TO DISTRIBUTION**

The exclusive right of reproduction is complemented by an exclusive right of distribution in the Computer Program Directive article 4(c): ‘. . . any form of distribution to the public, including the rental, of the original computer program or of copies thereof’. The typical situation is that copies are offered for sale across the counter, and this is obviously covered by the clause. The exclusive right requires that each copy of the program has to be distributed with the consent of the right holder.

Outside the consumer market, programs may be offered on the understanding that they will be configured for the individual user – this will also be a distribution of copies. ‘Rental’ is explained in recital 16 as ‘the making available for use, for a limited period of time and for profit-making purposes, of a
computer program or a copy thereof’. This will include the arrangement where the licence to use a program relies on periodic payments, often combined with a ‘maintenance contract’ for keeping the program current with respect to the developments of hardware and other relevant programs. There is a Swedish case relating to Nintendo gaming modules, where these were sold with an offer of repurchase of the game from the customer on favourable terms. This was not seen as an infringement of the distribution right.\(^{21}\)

The Computer Programs Directive makes no mention of public lending, which remains outside the scope of the Directive (see recital 16). Article 3 of the Rental and Lending Right Directive\(^{22}\) explicitly states that this Directive is without prejudice to article 4(c) of the Computer Program Directive. On the other hand, it is only this provision which is excluded. In article 5(2) of the Rental and Lending Right Directive there is a provision applying to the situation where the exclusive lending right is not implemented in national legislation for computer programs; in that case, remuneration for authors should be introduced.

The provision is cast on the basis of computer programs being made available to the market on some sort of carrier – diskettes or compact disks being typical examples. Currently, programs are typically acquired by downloading the program from a site. In this case, the purchase does not presume distribution. Downloading will result in a copy being made on the computer of the purchaser, but this copy has not been physically handed over from the right holder. The purchase through downloading is an example of a service (an ‘information society service’ in the terminology of the Electronic Commerce Directive).\(^{23}\) The result of the downloading obviously is a reproduction falling within the exclusive right according to the Computer Programs Directive.

**EXHAUSTION**

Article 4(c) of the Computer Programs Directive establishes the principle of regional exhaustion of the distribution right. It applies when a copy of a computer program is *sold*; the distribution right in this copy is then exhausted.

The copy has to be subject to a sale. There may be other transactions which make a copy of a computer program available to a party. Typically, there may be a licence agreement which makes the program available for the period a

---


\(^{22}\) Council Directive 92/100/EEC of 19 November 1992 on rental right and lending right and on certain rights related to copyright in the field of intellectual property. This Directive has a somewhat broader definition of ‘rental’; see art. 1(2).

licence fee is being paid. This will not constitute the sale of a copy of the program, but the qualification of the transaction may not be trivial, and will have to take the details of the situation into consideration.

Exhaustion only takes place within the European Community, and the European Economic Area. This is the general principle of exhaustion promoted by the EU, and a discussion of this in general falls outside the scope of this brief introduction.

Exhaustion includes only the copy subject for sale. In the case of off-the-shelf software, it only includes the original reproduction of the program on the carrier, which typically will be a compact disk.

In some instances, a computer is purchased with some pre-installed programs, typically operating system and some basic office programs. In this case, the copy has been purchased residing on the hard drive of the computer, and the distribution right is exhausted in these copies by the purchase, permitting the computer to be re-sold with the same programs. However, there are complicating factors, as such programs may be updated through online services, and if the updates considered separately qualify as works (as often will be the case for major updates), they have not been subject to sale as copies, and they are the result of services made available through the net.

This is an indication of the exhaustion principle not being quite straightforward. As mentioned, exhaustion does not apply to a program downloaded from a site. Occasionally, a hard copy in the form of a compact disk is offered as a back-up measure, being mailed to the purchaser separately, and typically for an additional fee. The distribution right in this must be subject to exhaustion, and may probably be sold by the purchaser to a third party without the consent of the right holder, while the purchaser continues to enjoy the advantages of the downloaded copy. To avoid this, the right holder may employ some technological protection measures.

Taking the typical example where off-the-shelf programs are purchased, the distribution right in the original copy will be exhausted. The purchaser may therefore sell this to a third party. But the right of reproduction is not exhausted. Therefore, the third party – who has no contractual relationship with the right holder – does not necessarily enjoy the limitations of these exclusive rights set out in article 5 of the Computer Programs Directive. For instance, if the third party has purchased a compact disk with a program and wants to use this on his or her computer, the program has to be uploaded to the hard disk of the computer, which constitutes a reproduction. Without a contract, it may be queried whether article 5(1) of the Computer Programs Directive applies, as this provision may be overruled by contractual arrangement. However, in the case of resale, the right holder has no possibility of making such arrangements.

As briefly mentioned, some implementations have made the provision of
the Computer Program Directive part of the general limitation of exclusive rights rather than provisions on the interpretation of contracts. If this strategy has been used, the problem indicated above is overcome, as a third party acquiring a copy of the program will be able to claim the same limitations in the right of reproduction.

But otherwise, one should interpret article 5 of the Computer Programs Directive to apply also to a third party having acquired a copy of the program in which the distribution right has been exhausted. It may be slightly at odds with conventional copyright principles to construe the transfer of a licence for reproduction with the copy in which the distribution right has been exhausted. But otherwise it would seem that the exhaustion to a large degree would be meaningless; one would be permitted to transfer the copy to a third party without the consent of the right holder, but that third party would not be in a position to make the intended use of the copy without such consent.

Decompilation

The right for decompilation is an innovation in copyright law. Its background may be found in the 1980s dispute between IBM and the European Commission, which was not dissimilar to the dispute with Microsoft which ended in 2007. In both instances the conflict was oriented towards a dominant enterprise, and its use of market power.

Under copyright law, certain exclusive rights are granted to the right holder. When the copyrighted work is made available on the market, anyone can avail themselves of the content, learn from the work and be inspired by it – and on this basis go on to create new and independent works by themselves. The exclusive right securing the position of the right holder is traded in the knowledge to be gained from the protected works. For computer programs, this (simplified) justification does not work – the programs are typically marketed in object form, which in practice cannot be read by a human.

Computer programs work in a context. When developing a program for a certain function, the developer would like to draw upon the functions of another program. One may think of a simple example, where a program is developed to assist translation from Norwegian to English. By highlighting a term, the auxiliary program will look for translations from its database. In order to do this, the developer of the auxiliary program needs to know how to communicate with the word processing program. This is often seen as a question of interface between a program and the surrounding domain. With knowledge of the interface, other programs may communicate with the program in the way defined by the program, and one may obtain interoperability between programs.

Interoperability is seen as desirable; it fosters competition and stimulates growth. Therefore it is seen as desirable for providers of a program to publish the specification of the interface of a program. But the provider of a program
may be reluctant to do so, because ignorance of the interface may offer some competitive protection.

This is the issue which article 6 of the Computer Programs Directive attempts to solve. It was easily the most controversial provision of the Directive, and the provision is rather complex. In this context, it will not be discussed in all its details, but an attempt is made to outline its structure.

First, the provision may only be applied by someone developing an independent program. It is the situation of the developer which may trigger the right of decompilation. The developer sees interoperability with another program as desirable. If necessary data on the interface is published or otherwise available, there is no need to look further, and the right to decompilation cannot be invoked. Decompilation may take place only when it is ‘indispensable’.

If decompilation is indispensable, further conditions apply. The program to be analysed must be lawfully available, the decompilation can only be performed by a licensee or another having the right to use a copy of the program and the decompilation can only be made of ‘the parts of the original program which are necessary to achieve interoperability’. However, as the original program is only available in object form, it may be difficult for the person doing the decompilation to determine which parts are necessary: only what is available can be used to determine what to decompile.

The result of the decompilation can only be used for gaining the necessary interoperability, and not for any other purpose. The provision emphasises that decompilation does not authorise development, marketing etc. of a program ‘substantially similar in its expression’ or any other act which infringes copyright. To drive this point home, article 6(3) of the Computer Program Directive refers to the Berne Convention, and emphasises that the application of decompilation cannot be ‘used in a manner which unreasonably prejudices the right holder’s legitimate interests or conflicts with a normal exploitation of the computer program’.

Decompilation is therefore construed as a rather narrow right. If data on interfacing with a program is available to developers, it will not apply. And when applying, it only applies to the extent necessary to establish the desired interoperability with a program independently created.

In practice, decompilation results in a reproduction of the program, the object code being transformed to a form of source code by the strict rules of the decompiling program. As noted above, it will not be identical to the original source code due to the processing by the programs performing the compilation

---

24 Though there will be rumours that the published data do not specify the most efficient way to interact with the program, etc.
and decompilation. The resulting ‘pseudo source code’ may be quite a jumble, and require considerable skill and time to analyse. Therefore, decompilation may be less of a solution than seen at first glance. The major effect of the provision is that on legal policy: to encourage the publication of interface specification and through this, exclude the application of the provision.

**Technical protection measures**

According to article 7(c) of the Computer Programs Directive, special protection is established for technical protection measures. The discussion of digital right management and technical protection measures became very heated with respect to the Copyright Directive, which in its article 6 implements article 11 of the WIPO Copyright Treaty (1996) on technological measures. The provision of the Computer Program Directive predates this provision, and is excluded from the Copyright Directive (see article 1(2)(a)). Therefore, the provision on technical protection measures for computer programs applies in parallel to the more extensive regulation in the Copyright Directive.

At the time the Computer Programs Directive was passed, such protection devices were less sophisticated than examples currently being deployed. A typical device would be a plug (a ‘dongle’) having to be present at the connection between the computer and printer. The program would check for its presence; if not found, the program would fail to print. This gave some security for the program being unlawfully reproduced as the plug itself was difficult to duplicate. Other measures would be key diskettes having to be present when initiating the program, the use of codes, etc.

The Computer Programs Directive applies to the possession, or putting into circulation of ‘any means’ which facilitate ‘the unauthorized removal or circumvention of any technical device which may have been applied to protect a computer program’. The act of circumvention itself is not qualified as illegal; the provision focuses on the ‘means’. The term ‘means’ is not specified, but will probably include ‘devices, products or components’ (see article 6(2) of the Copyright Directive), but probably not services. The subject for

---

25 See also art. 18, WPPT.
26 In a German case, the court held that as the program was sold with a dongle, the intended use also presumed the use of the dongle, referring to the delimitation of the restricted acts under art. 5. It might also have been discussed in the context of art 7(c). See Oberlandesgericht Karlsruhe 1 October 1996, 6 U 40/95.
27 In a Finnish case in which written instructions on how to circumvent were available, the Supreme Court did not find that this qualified as putting into circulation a means for circumvention according to the Finnish implementation of the Computer Programs Directive, see *Adobe Systems Inc v. [A] Software Distribution* [2004] ECDR (30) 303.
circumvention is qualified as a ‘technical device’. It is suggested that this is hardly a limitation – any measure implemented for a computerised system will be ‘technical’ in some sense of the term, but the term ‘device’ would seem – as suggested above – to exclude services.

There are two conditions. First, this must be the ‘sole intended purpose’ of the prohibited device. This is more limited than in the later Copyright Directive. Many devices may permit circumvention, but they may also have other lawful applications, for instance to translate formats. Second, the act must be for ‘commercial purposes’. This will exclude acts which are done by private persons – circumvention protection devices to access a program without any commercial purpose.

**Term of protection**

Article 8 of the Computer Program Directive stipulates the term of protection. This was originally fifty years counted from – as for other works – alternative dates, the major one being the death of the author (*post mortem auctoris*). However, the period was amended by article 1(1) of the Term Directive for all literary works, including computer programs, to seventy years.

In this context, there is little reason to discuss this provision further, perhaps only with a small note on whether such a relatively long term of protection is justified for computer programs. The argument may be that though the term of protection may be justified for other types of literary works (and here opinions differ), it is not justified for works which are seen as rather short lived.

But the brief life-span of computer programs is to some extent a myth. Computer technology has since the beginning been subject to rapid development, which makes equipment obsolete in a matter of years. It is fascinating to note that the programs often survive through these changes by amendments and functional enhancement. Through incremental development, programs will make the leap from one generation of hardware to the next. There is no lack of examples of programs originally designed for the first personal computers in the early 1980s which still have a place in the market or, indeed, programs designed for managing the databases made possible by the first large magnetic discs still making their imprint on current systems in public administration.

---

28 See art. 6(3), Copyright Directive.
29 This is discussed in *Kabushiki Kaisha Sony Computer Entertainment Inc et al v. Ball et al*, [2004] EWHC 1738 (ChD) with respect to the ‘Messiah2’ chip, where Justice Laddie rejected that the chip had any other purpose than to circumvent the copy protection of Sony Playstation 2.
To unravel the term of protection in such cases must in practice be difficult. One will be able to determine when the term starts for the first version of a program. When the next version comes along, one has to decide whether the amendments meet the test of originality and represent a derivative work, or whether only changes of no copyright relevance have taken place. In the first case, a new term of protection will be initiated for the derivative work. This will happen time and again. The original coding may get lost in the chain of development – in principle there may be old versions actually passing into the public domain, but in practice this will be difficult to determine and of little practical importance.

The term of protection is, however, sufficiently long for us not to have had any direct experience. What is generally accepted to be the first electronic computer, the ENIAC, was unveiled in 1946, and was programmed by manipulating its switches and cables. Only when the von Neumann architecture was adopted for storing the program in logical notation separate programs could be written – the first stored program for the amended ENIAC was initiated 16 September 1948. This program was written for John von Neumann by Adele Goldstine, who died in 1964. Of course, other programmers would have died before her, but it may be an appropriate note of curiosity on which to close this introduction to reflect on the fact that the term of protection of the first program will last till 2035.